**ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**
**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**
**КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ**

# КВАЛІФІКАЦІЙНА РОБОТА

на тему: «СИСТЕМА РОЗПІЗНАВАННЯ МАТЕМАТИЧНИХ ВИРАЗІВ ЗАСОБАМИ CNN-МОДЕЛЕЙ»

на здобуття освітнього ступеня магістра
зі спеціальності 123 Комп'ютерної інженерії
*(код, найменування спеціальності)*
освітньо-професійної програми Комп'ютерні системи та мережі
*(назва)*

*Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело*

_____ *Владислав СЛЮСАР*
*(підпис)* *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КСДМ-61
Владислав СЛЮСАР
Ім'я, ПРІЗВИЩЕ

Керівник: Кирило СОКОЛОВ
*науковий ступінь, вчене звання* Ім'я, ПРІЗВИЩЕ

Рецензент: _____
*науковий ступінь, вчене звання* Ім'я, ПРІЗВИЩЕ

**Київ 2025**

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**
**Навчально-науковий інститут Інформаційних технологій**

Кафедра 123 Комп'ютерної інженерії
Ступінь вищої освіти Магістр
Спеціальність 123 Комп'ютерної інженерії
Освітньо-професійна програма Комп'ютерні системи та мережі

**ЗАТВЕРДЖУЮ**
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
Ім'я, ПРІЗВИЩЕ
«_____» _____ 2025р.

# ЗАВДАННЯ
# НА КВАЛІФІКАЦІЙНУ РОБОТУ

Слюсар Владислав Сергійович
*(прізвище, ім'я, по батькові здобувача)*

1. Тема кваліфікаційної роботи: Система розпізнавання математичних виразів засобами CNN-моделей
керівник кваліфікаційної роботи Кирило СОКОЛОВ,
*(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)*
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «30» 10 2025р. № 467
2. Строк подання кваліфікаційної роботи «_____» _____ 2025р.
3. Вихідні дані до кваліфікаційної роботи: Науково-технічна література, методи глибокого навчання, комп'ютерного зору, системного аналізу для проектування архітектури ПЗ та методи статистичної оцінки якості моделей.
Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Теоретичні аспекти застосування згорткових нейронних мереж для обробки символьної інформації
2. Аналітичний огляд сучасних програмних рішень та методів розпізнавання
3. Методика практичної реалізації нейромережевої моделі та аналіз отриманих результатів
4. Перелік ілюстративного матеріалу: *презентація*
5. Дата видачі завдання «_____» _____ 2025р.

# КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Збір даних | 25.09-24.10.2025 | Виконано |
| 2 | Обробка матеріалу | 25.10-05.11.2025 | Виконано |
| 3 | Розробка теоретичної частини | 06.11-10.11.2025 | Виконано |
| 4 | Розробка системи | 11.11-15.11.2025 | Виконано |
| 5 | Висновки за результатами аналізу | 15.11-19.11.2025 | Виконано |
| 6 | Оформлення пояснювальної записки | 19.11-21.11.2025 | Виконано |
| 7 | Розробка презентації | 21.11-29.11.2025 | Виконано |
| 8 | Передзахист | 01.12-02.12.2025 | Виконано |
| 9 | Здача в деканат | 15.12-18.12.2025 | Виконано |

Здобувач вищої освіти _____ Владислав СЛЮСАР
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Кирило СОКОЛОВ
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

# РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 61 стор., 37 рис., 20 джерел.

*Мета дослідження* – створення ефективної програмної системи для автоматичного детектування та розпізнавання математичних формул і символів на зображеннях із використанням методів глибокого навчання.

*Об'єкт дослідження* – процеси обробки та розпізнавання візуальної інформації, що містить математичну нотацію.

*Предмет дослідження* – архітектури згорткових нейронних мереж та методи підготовки даних для задач детекції математичних символів.

*Короткий зміст роботи*: У кваліфікаційній роботі проведено порівняльний аналіз архітектур CNN та RNN, обґрунтовано вибір однопрохідного детектора YOLOv5. Реалізовано двоетапну схему роботи: локалізація формули на зображенні та подальша детекція окремих символів. Здійснено збір, розмітку та очищення навчальних даних з використанням платформи Roboflow. Навчено дві нейромережеві моделі та розроблено програмне забезпечення на мові Python, яке виконує оркестрацію процесу розпізнавання, фільтрацію помилкових детекцій та взаємодію з користувачем.

Результати: Розроблена система демонструє високу точність локалізації формул mAP 0.983 та розпізнавання символів mAP 0.898. Створено власний верифікований набір даних. Запропоновано алгоритми пост-обробки для впорядкування розпізнаних символів.

КЛЮЧОВІ СЛОВА: ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, YOLOv5, КОМП'ЮТЕРНИЙ ЗІР, РОЗПІЗНАВАННЯ ФОРМУЛ, ГЛИБОКЕ НАВЧАННЯ, DATASET, ROBOFLOW, DETECTFORMULA.

# ABSTRACT

Textual part of the master's qualification paper: 61 pages, 37 figures, 20 sources.

The purpose of the study is to create an effective software system for the automatic detection and recognition of mathematical formulas and symbols on images using deep learning methods.

The object of the study is the processes of processing and recognizing visual information containing mathematical notation.

The subject of the study is the architectures of convolutional neural networks and data preparation methods for mathematical symbol detection tasks.

Brief content of the paper: The qualification paper conducts a comparative analysis of CNN and RNN architectures and justifies the choice of the single-shot detector YOLOv5. A two-stage workflow is implemented: formula localization on the image followed by the detection of individual symbols. Data collection, annotation, and cleaning were performed using the Roboflow platform. Two neural network models were trained, and Python-based software was developed to perform recognition process orchestration, false detection filtering, and user interaction.

Results: The developed system demonstrates high accuracy in formula localization mAP 0.983 and symbol recognition mAP 0.898. A proprietary verified dataset was created. Post-processing algorithms for ordering recognized symbols were proposed.



KEYWORDS: CONVOLUTIONAL NEURAL NETWORK, YOLOv5, COMPUTER VISION, FORMULA RECOGNITION, DEEP LEARNING, DATASET, ROBOFLOW, DETECTFORMULA.

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ**
**ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

**Навчально-науковий інститут Інформаційних технологій**

# ПОДАННЯ
# ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
# ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
**на здобуття освітнього ступеня магістра**

Направляється здобувач Слюсар В. С. до захисту кваліфікаційної роботи
*(прізвище та ініціали)*
за спеціальністю 123 Комп'ютерна інженерія
*(код, найменування спеціальності)*
освітньо-професійної програми Комп'ютерні системи та мережі
*(назва)*
на тему: «Система розпізнавання математичних виразів засобами CNN-моделей».
Кваліфікаційна робота і рецензія додаються.

Директор ННІ _____ Катерина НЕСТЕРЕНКО
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

**Висновок керівника кваліфікаційної роботи**

Здобувач Слюсар В. С.показав відмінну теоретичну підготовку, уміння володіти та розуміння новими комп'ютерними технологіями, користуватися навчальною і науково-технічною літературою. Працюючи над завданнями, які були поставлені керівником, проявив ініціативність, сумлінність та хист до інженерної роботи.
Все це дозволяє оцінити кваліфікаційну роботу студента Слюсара Владислава Сергійовича на оцінку «відмінно», та присвоїти їй кваліфікацію «магістр з комп'ютерної інженерії».

Керівник кваліфікаційної роботи _____ Кирило СОКОЛОВ
*(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

«_» _____ 2025 року

**Висновок кафедри про кваліфікаційну роботу**

Кваліфікаційна робота розглянута. Здобувач *Слюсар В. С.* допускається до захисту даної роботи в Екзаменаційній комісії.

Завідувач кафедрою Комп'ютерної інженерії _____ Наталія ЛАЩЕВСЬКА
*(назва)* *(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

**ВІДГУК РЕЦЕНЗЕНТА**
**на кваліфікаційну магістерську роботу**

здобувача вищої освіти Слюсар Владислав Сергійович
*(прізвище, ім'я, по батькові)*
на тему: «Система розпізнавання математичних виразів засобами CNN-моделей»

**Актуальність:**

Кваліфікаційна магістерська робота присвячена вирішенню затребуваного завдання в галузі штучного інтелекту — автоматизації введення та обробки математичних текстів. Зважаючи на зростаючу потребу в цифровізації наукового контенту та розвитку EdTech-рішень, тема розпізнавання складних графічних структур, якими є формули, є вкрай актуальною. Робота пропонує сучасний підхід із використанням швидкісних нейронних мереж сімейства YOLO, що дозволяє виконувати обробку в режимі реального часу, на відміну від класичних повільних OCR-систем.

**Позитивні сторони:**
1. Автор провів ґрунтовну роботу з підготовки даних, що є найбільш трудомістким етапом у задачах Computer Vision. Створення, ручна розмітка та верифікація понад 9000 об'єктів свідчить про високий рівень практичної залученості дослідника.
2. У роботі наведено детальний аналіз теоретичної бази, зокрема порівняння CNN та RNN , а також обґрунтовано вибір конкретної модифікації YOLOv5s на основі балансу швидкості та точності.
3. Вдалим архітектурним рішенням є розділення задачі на два послідовні етапи. Це дозволило значно зменшити вплив шумів та підвищити точність класифікації дрібних символів, що підтверджується високими метриками.

**Недоліки:**
1. У роботі основна увага приділена детектуванню друкованих символів та формул. Питання розпізнавання рукописного тексту, хоч і згадується в огляді аналогів та прикладах, потребує більш детального аналізу результатів точності саме на рукописних даних, які мають значно вищу варіативність.
2. Опис алгоритму відновлення структурних зв'язків описано на рівні загальної логіки пост-обробки. Бажано було б навести більш формалізований математичний опис правил просторового аналізу взаємного розташування bounding box-ів.

**Висновок:** *кваліфікаційна бакалаврська робота заслуговує оцінку "відмінно", а здобувач Слюсар В. С. заслуговує присвоєння кваліфікації: магістр з комп'ютерної інженерії.*

Рецензент: _____ _____
*науковий ступінь, вчене звання* *підпис* *Ім'я, ПРІЗВИЩЕ*

**ЗМІСТ**

ВСТУП.....9
1 ТЕОРЕТИЧНІ АСПЕКТИ ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ОБРОБКИ СИМВОЛЬНОЇ ІНФОРМАЦІЇ.....11
1.1 Архітектура та принципи функціонування згорткових нейронних мереж 11
1.2 Особливості застосування одновимірних згорток та методи векторизації символьної інформації.....15
1.3 Порівняльний аналіз ефективності CNN у задачах обробки природної мови порівняно з рекурентними мережами.....18
2 АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ПРОГРАМНИХ РІШЕНЬ ТА МЕТОДІВ РОЗПІЗНАВАННЯ.....22
2.1 Аналіз функціональних можливостей програмного комплексу Microsoft Math Solver. .....22
2.2 Огляд функціональних можливостей та алгоритмів роботи системи PhotoMath .....25
2.3 Особливості реалізації рукописного введення та розпізнавання формул у середовищі Microsoft OneNote.....27
3 МЕТОДИКА ПРАКТИЧНОЇ РЕАЛІЗАЦІЇ НЕЙРОМЕРЕЖЕВОЇ МОДЕЛІ ТА АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ.....31
3.1 Обґрунтування вибору та архітектурні особливості нейронної мережі YOLOv5.....31
3.2 Методика підготовки та анотування графічних даних для навчання нейронних мереж.....40
3.3 Структура набору даних та методика навчання нейронної мережі .....45
3.4 Тестування та верифікація роботи нейронної мережі.....51
ВИСНОВКИ.....55
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....57
Додаток А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація).....60

# ВСТУП

У сучасному цифровому просторі автоматизована обробка документів набуває критичного значення, проте розпізнавання математичних текстів залишається одним із найскладніших викликів у галузі комп'ютерного зору. На відміну від звичайного тексту, математичні вирази мають складну двовимірну структуру, де просторове розміщення символів (індекси, ступені, дроби) несе важливе семантичне навантаження. Традиційні методи оптичного розпізнавання символів, що працюють лінійно, часто виявляються неефективними для коректної інтерпретації таких даних.

Це зумовлює необхідність переходу до використання методів глибокого навчання, зокрема згорткових нейронних мереж, які здатні автоматично виявляти складні патерни та ієрархію ознак безпосередньо із зображення. Впровадження сучасних архітектур, таких як YOLO, дозволяє вирішувати задачу детектування математичних об'єктів у реальному часі з високою точністю, що відкриває нові можливості для цифровізації наукового та навчального контенту.

Мета і завдання дослідження. Метою роботи є розробка програмної системи для автоматизованого розпізнавання математичних формул на зображеннях із використанням згорткових нейронних мереж, що забезпечує високу точність детектування та класифікації символів.

Для досягнення мети поставлені наступні завдання:

1. Проаналізувати теоретичні аспекти функціонування згорткових нейронних мереж та порівняти їх ефективність із рекурентними мережами у задачах обробки символьної інформації.

2. Здійснити аналітичний огляд існуючих програмних рішень для визначення вимог до проектованої системи.

3. Обґрунтувати вибір архітектури нейронної мережі YOLOv5s та розробити двоетапну методику розпізнавання формула — символ.

4. Сформувати та підготувати спеціалізовані набори даних для навчання моделей, провівши їх розмітку, очищення та аугментацію.

5. Виконати навчання двох екземплярів нейронної мережі для задач локалізації формул та класифікації окремих символів, оптимізувавши їх гіперпараметри.

6. Розробити модульну архітектуру програмного застосунку, що включає механізми пост-обробки та інтерфейс взаємодії з користувачем.

Об’єкт дослідження – процес автоматизованого розпізнавання візуальних образів математичних виразів

Предмет дослідження – методи та алгоритми згорткових нейронних мереж сімейства YOLO для детектування та класифікації математичних символів.

Методи дослідження. У роботі використано методи глибокого навчання, комп'ютерного зору, системного аналізу для проектування архітектури ПЗ та методи статистичної оцінки якості моделей.

Практичне значення. Розроблена система дозволяє автоматизувати процес переведення математичних формул з графічного формату у машиночитаний вигляд, що може бути використано в освітніх платформах та системах документообігу.

# 1 ТЕОРЕТИЧНІ АСПЕКТИ ЗАСТОСУВАННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ОБРОБКИ СИМВОЛЬНОЇ ІНФОРМАЦІЇ

## 1.1 Архітектура та принципи функціонування згорткових нейронних мереж

Згорткові нейронні мережі (ЗНМ) на сьогоднішній день представляють собою окремий, спеціалізований клас штучних нейронних мереж, який фактично став домінуючим стандартом у вирішенні широкого спектру завдань комп'ютерного зору та розпізнавання образів. Цей тип архітектур викликає значний науковий та практичний інтерес у різних галузях промисловості та науки завдяки своїй високій ефективності. Головною особливістю та перевагою ЗНМ є їхня здатність до автоматичного та адаптивного вивчення просторових ієрархій ознак безпосередньо з вхідних даних. Цей процес навчання реалізується за допомогою алгоритму зворотного розповсюдження помилки та базується на послідовному використанні кількох типів функціональних блоків, які, поєднуючись у складну структуру, дозволяють моделювати процеси зорового сприйняття. До основних структурних елементів таких мереж відносять шари згортки, шари субдискретизації (або об'єднання) та повнозв'язкові шари, кожен з яких виконує свою унікальну роль у перетворенні вхідного сигналу на вихідний результат класифікації або розпізнавання.

Фундаментальним компонентом архітектури ЗНМ є згортковий шар, оскільки саме на нього покладається основне обчислювальне навантаження всієї системи. Робота цього шару базується на математичній операції згортки, яка передбачає взаємодію кількох ключових складових: масиву вхідних даних, набору фільтрів (ядер згортки) та результуючих карт ознак. Процес обробки інформації на цьому етапі можна описати як ковзання фільтра по всій площі

вхідного зображення з певним кроком. У кожній позиції фільтра відбувається обчислення скалярного добутку між значеннями фільтра та відповідною локальною ділянкою вхідних даних, до якого додається значення зміщення. Схематично принцип роботи згорткового шару та процес формування карти ознак зображено на рис.1.1. Результатом цієї операції є формування карти ознак, яка відображає наявність певних візуальних патернів у різних частинах зображення. Варто зазначити, що кожен фільтр відповідає за виділення конкретної характеристики, наприклад, ліній, кутів або кольорових переходів. Принцип дії згорткового шару наочно демонструє механізм локального сприйняття інформації, що є критично важливим для ефективного аналізу зображень.

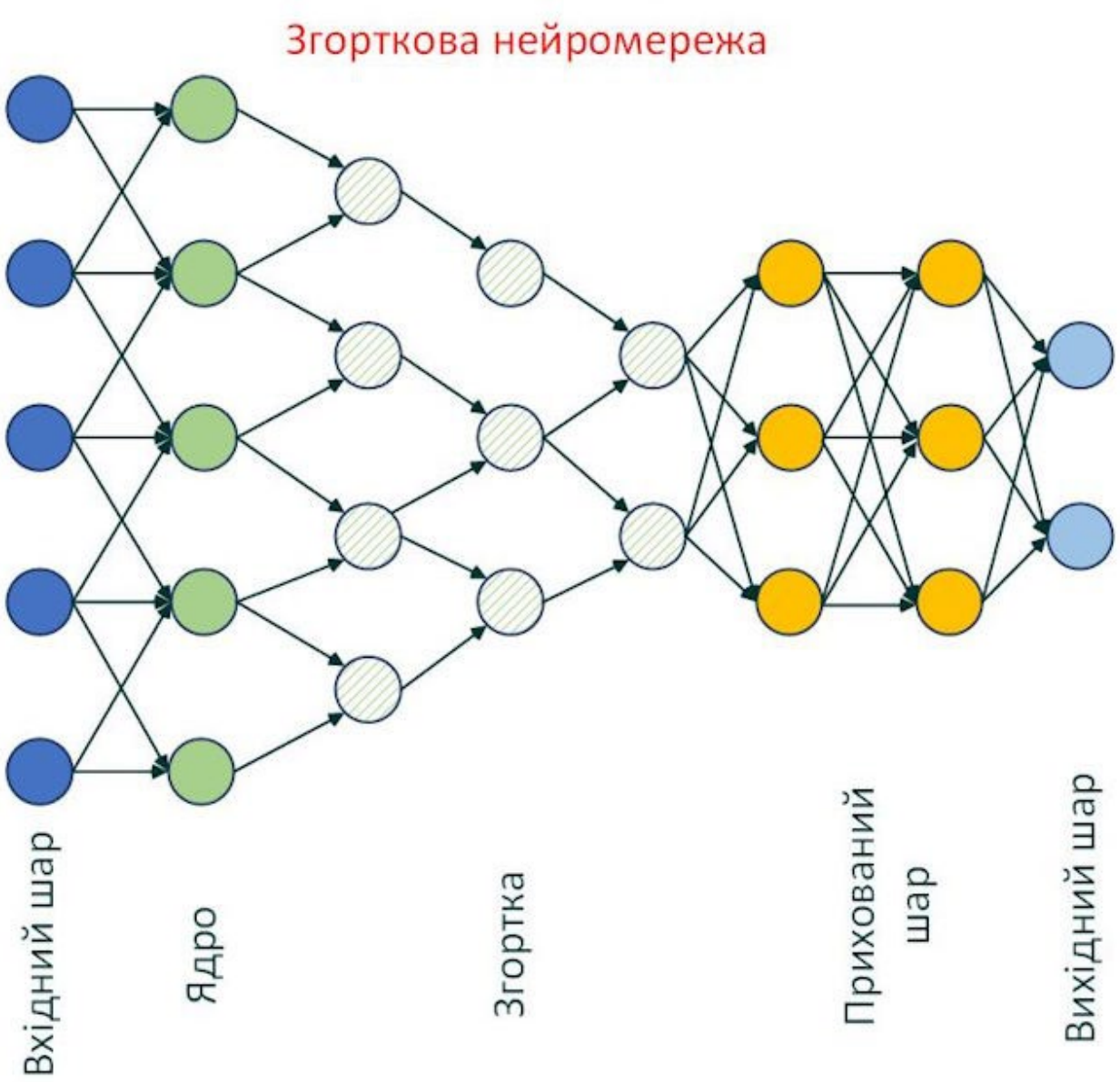



Рисунок 1.1 — Схема роботи згорткового шару

Наступним важливим елементом архітектури є об'єднуючий шар, який також часто називають шаром субдискретизації або пулінгу. Його головне призначення полягає у зниженні просторової розмірності карт ознак, що дозволяє зменшити загальну кількість параметрів моделі та обсяг необхідних обчислень у наступних шарах. Подібно до згорткового шару, операція об'єднання передбачає проходження вікна фільтра по вхідних даних, проте існує суттєва відмінність у механізмі обробки. Фільтр об'єднуючого шару не містить вагових коефіцієнтів, які потрібно навчати. Замість цього він використовує фіксовану функцію агрегування значень у межах свого

рецептивного поля. Найпоширенішими методами агрегування є вибір максимального значення або обчислення середнього арифметичного серед значень, що потрапили у вікно фільтра. Результатом роботи цього шару є заповнення вихідного масиву зменшеної розмірності, що також сприяє підвищенню інваріантності моделі до невеликих зміщень та спотворень вхідного об'єкта.

Завершальним етапом обробки інформації в типовій архітектурі згорткової нейронної мережі є проходження даних через повнозв'язкові шари. Назва цього типу шарів повністю відображає принцип їхньої організації, оскільки кожен нейрон такого шару має безпосередній зв'язок з усіма нейронами попереднього рівня. Це суттєво відрізняє їх від згорткових шарів, де нейрони пов'язані лише з локальними рецептивними полями. Якщо на попередніх етапах мережа виділяла та ущільнювала локальні ознаки зображення, то завданням повнозв'язкового шару є глобальний аналіз усієї сукупності отриманих характеристик. Саме тут відбувається інтеграція всієї добутої інформації для прийняття остаточного рішення.

Цей шар виконує фінальне завдання класифікації на основі високорівневих ознак, які були успішно екстраговані та трансформовані попередніми згортковими та об'єднуючими шарами. Важливу роль у функціонуванні нейронної мережі відіграють функції активації, які вносять необхідну нелінійність у процес обробки даних. У той час як у згорткових шарах та шарах субдискретизації загальноприйнятим стандартом є використання функції випрямленого лінійного вузла (ReLU) для прискорення навчання та уникнення проблеми згасання градієнта, у вихідному повнозв'язковому шарі підхід змінюється. Для вирішення завдань класифікації тут зазвичай застосовуються інші функції активації, такі як софтмакс або сигмоїда, які дозволяють трансформувати вихідні значення нейронів у розподіл ймовірностей. Це дає можливість інтерпретувати вихід мережі як оцінку ймовірності приналежності вхідного зображення до того чи іншого класу в діапазоні від нуля до одиниці.

## 1.2 Особливості застосування одновимірних згорток та методи векторизації символьної інформації

Використання згорткових нейронних мереж (CNN) традиційно асоціюється з обробкою зображень, де двовимірні фільтри сканують площину пікселів для виявлення візуальних патернів. Однак у задачах обробки природної мови (NLP), де дані мають послідовну структуру, застосовуються одновимірні згортки (1D-CNN). Ключова відмінність полягає у розмірності вхідних даних: якщо зображення є двовимірною матрицею, то текст представляється як послідовність векторів. У цьому контексті ядро згортки (фільтр) рухається лише в одному напрямку — вздовж часової осі або послідовності символів, охоплюючи всю розмірність вектора ознак на кожному кроці. Це дозволяє мережі фокусуватися на локальних залежностях між сусідніми елементами послідовності, ефективно виділяючи n-грами різного рівня складності без необхідності явного лінгвістичного аналізу.

Архітектурна перевага 1D-CNN перед рекурентними мережами (RNN або LSTM) полягає у можливості паралелізації обчислень. Оскільки застосування фільтра до одного фрагмента тексту не залежить від результату обробки попереднього фрагмента, весь масив даних може оброблятися одночасно. Це суттєво пришвидшує процес навчання та інференсу, особливо на довгих послідовностях. Згортковий шар формує нову карту ознак, де кожен елемент є результатом зваженої суми вхідних векторів у межах вікна згортки, пропущеної через функцію активації. Важливим етапом після згортки є операція підвибірки (pooling), найчастіше max-pooling. Вона виконує функцію зниження розмірності та виділення найбільш значущих ознак, роблячи модель інваріантною до незначних зсувів у вхідних даних. Тобто, якщо ключова фраза або патерн з’явиться в іншій частині речення, мережа все одно зможе її ідентифікувати як ту саму ознаку, рис.1.2.

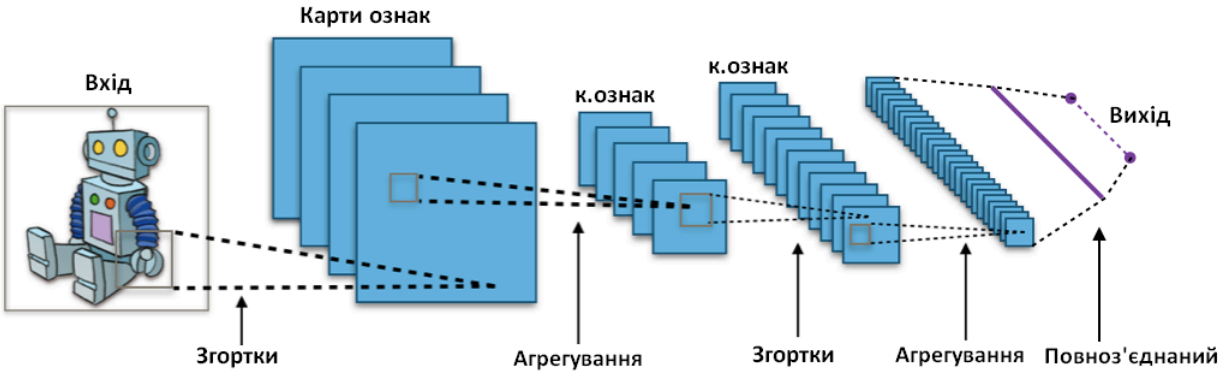



Рисунок 1.2 — Принцип дії одновимірної згортки при обробці послідовних даних: проходження ядра фільтра вздовж часової осі та формування карти ознак

Коли мова йде про символьну обробку тексту (Character-level CNN), вхідними даними стають не слова, а окремі символи. Такий підхід радикально змінює парадигму векторизації. Традиційні методи, що базуються на словниках слів, страждають від проблеми розрідженості даних та неможливості обробки слів, яких не було у навчальній вибірці (Out-of-Vocabulary problem). Натомість символьний підхід дозволяє працювати з будь-яким текстом, оскільки кількість унікальних символів у мові (алфавіт, цифри, спецсимволи) є обмеженою та заздалегідь відомою. Це робить модель надзвичайно стійкою до орфографічних помилок, друкарських одруків, сленгу та особливостей аглютинативних мов, де нові слова утворюються шляхом додавання суфіксів та префіксів. Мережа навчається розпізнавати морфологічну структуру слова безпосередньо з послідовності літер.

Першим етапом роботи з символьною інформацією є її векторизація, тобто перетворення дискретних символів у числовий формат, придатний для обробки нейромережею. Базовим методом є One-Hot Encoding (унітарне кодування). У цьому випадку кожен символ алфавіту асоціюється з унікальним вектором, довжина якого дорівнює загальній кількості символів у словнику. Всі елементи вектора є нулями, окрім одного, що відповідає індексу даного символу. Хоча цей метод є простим та інтуїтивно зрозумілим, він створює розріджені вектори високої розмірності, які не несуть інформації про

семантичну близькість символів. Крім того, відстань між будь-якими двома різними символами у такому просторі є однаковою, що не завжди відображає лінгвістичну реальність.

Ефективнішою альтернативою є використання шару навчуваних вкладень (Learned Embeddings). У цьому підході кожен символ відображається у щільний вектор дійсних чисел фіксованої розмірності, яка зазвичай значно менша за розмір словника. Значення у цьому векторі не задаються вручну, а є параметрами моделі, що оптимізуються під час навчання разом з вагами фільтрів згортки. Це дозволяє мережі самостійно знаходити закономірності та групувати схожі за функціональним призначенням символи у багатовимірному просторі ознак. Наприклад, голосні літери можуть отримати вектори, що знаходяться "ближче" один до одного, ніж до приголосних, або ж символи верхнього регістру можуть мати певні кореляції з відповідними символами нижнього регістру.

Поєднання шару символьних вкладень та одновимірних згорток створює потужну архітектуру, рис.1.3. Спочатку текст перетворюється на матрицю, де рядки відповідають позиціям символів у тексті, а стовпці — вимірам вектору вкладення. Потім 1D-згортка "ковзає" вздовж цієї матриці, охоплюючи кілька символів одночасно. Це дозволяє мережі на перших шарах вивчати прості комбінації літер (склади, закінчення), а на глибших шарах — формувати уявлення про слова та цілі фрази. Таким чином, модель автоматично вибудовує ієрархію ознак від найпростіших графічних елементів до складних семантичних конструкцій, не потребуючи попередньої токенізації чи лематизації тексту.

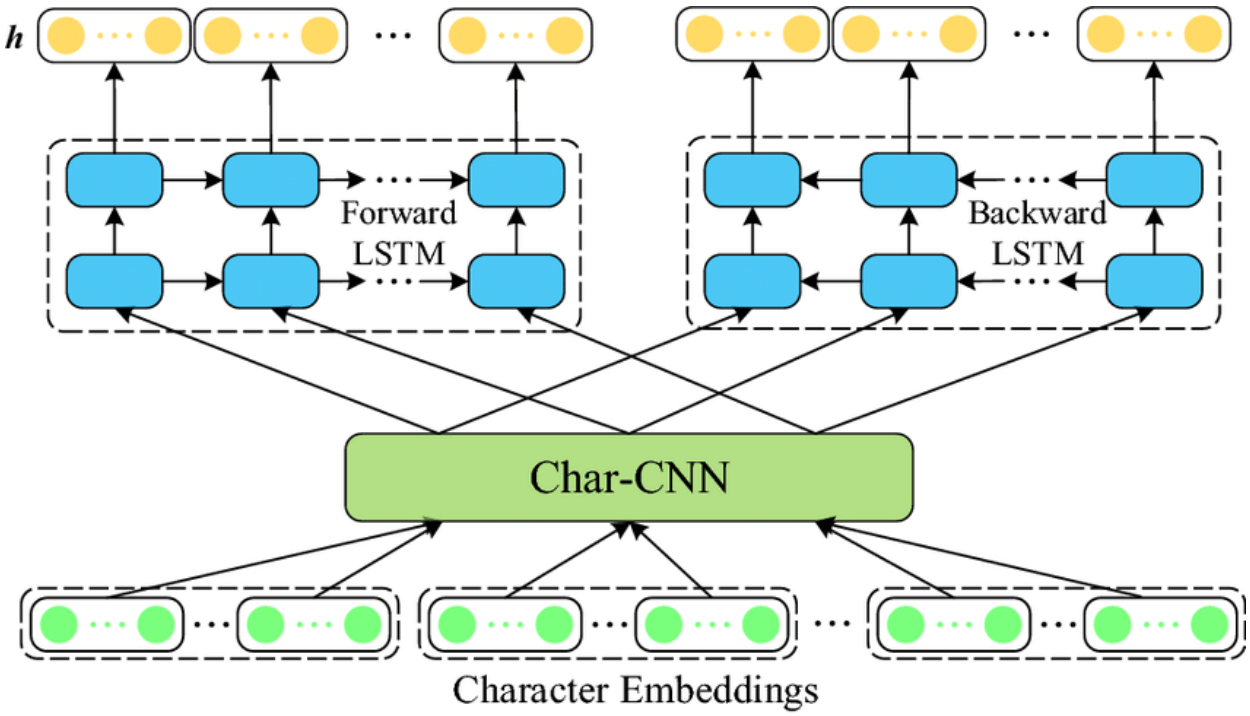



Рисунок 1.3 — Узагальнена структурна схема згорткової нейронної мережі символьного рівня (Char-CNN), що включає етапи вкладення, згортки, підвибірки та класифікації

Такий підхід до векторизації та обробки є особливо актуальним у задачах класифікації коротких текстів, аналізу тональності у соціальних мережах або детекції кібератак, де вхідні дані (наприклад, шкідливі URL-адреси або код) можуть містити навмисні спотворення, які б "зламали" класичні моделі на основі слів.

## 1.3 Порівняльний аналіз ефективності CNN у задачах обробки природної мови порівняно з рекурентними мережами

Еволюція методів глибокого навчання в обробці природної мови тривалий час характеризувалася домінуванням рекурентних нейронних мереж (RNN), зокрема їхніх модифікацій — LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Units). Ці архітектури інтуїтивно відповідали природі

людської мови, сприймаючи текст як часову послідовність, де значення поточного слова залежить від попереднього контексту. Однак поява та адаптація згорткових нейронних мереж (CNN) для текстових даних змінила парадигму, запропонувавши альтернативний підхід, що базується на виявленні локальних патернів. Порівняльний аналіз цих двох архітектур виявляє фундаментальні відмінності в обчислювальній ефективності, здатності до навчання та способах моделювання залежностей.

Ключова відмінність, що визначає перевагу CNN у швидкодії, полягає в механізмі обробки вхідних даних. Рекурентні мережі за своєю природою є послідовними: для обчислення прихованого стану на поточному кроці необхідно знати стан на попередньому кроці. Ця жорстка часова залежність унеможливлює розпаралелювання процесу навчання в межах одного прикладу (речення або документу), що призводить до неефективного використання ресурсів графічних процесорів (GPU) при роботі з довгими послідовностями. На противагу цьому, згорткові мережі не мають прихованого стану, що передається від слова до слова. Операція згортки застосовується до всіх фрагментів тексту одночасно та незалежно один від одного. Це дозволяє виконувати масивні паралельні обчислення, скорочуючи час навчання моделі на порядки, особливо при обробці великих корпусів даних.

З точки зору моделювання контексту, RNN теоретично здатні охоплювати необмежений контекст, зберігаючи інформацію про початок речення навіть в його кінці. Проте на практиці вони стикаються з проблемою зникаючого або вибухаючого градієнта, що ускладнює передачу сигналу помилки через довгий ланцюг часових кроків. Внаслідок цього рекурентні мережі часто "забувають" віддалений контекст, фокусуючись на останніх оброблених словах. CNN, у свою чергу, мають фіксоване рецептивне поле, розмір якого визначається шириною ядра згортки. Хоча один шар згортки бачить лише обмежений локальний контекст (наприклад, 3-5 слів), використання глибоких архітектур та операцій пулінгу дозволяє ієрархічно розширювати це поле. На верхніх шарах мережа отримує здатність бачити

глобальні залежності всього речення, уникаючи при цьому проблем з градієнтами, властивих RNN, рис.1.4.




Рисунок 1.4 — Структурне порівняння потоків даних: послідовна залежність станів у рекурентних мережах проти паралельної ієрархічної обробки у згорткових мережах

Важливим аспектом порівняння є специфіка розв'язуваних задач. Рекурентні мережі традиційно демонструють кращі результати в задачах, де критично важлива сувора послідовність та порядок слів, наприклад, у мовному моделюванні або машинному перекладі (до ери трансформерів). У таких випадках синтаксична структура та залежність слів одне від одного відіграють вирішальну роль. Натомість CNN виявляють вищу ефективність у задачах класифікації текстів, аналізу тональності та детекції спаму. Це пояснюється тим, що для класифікації часто достатньо виявити наявність певних ключових фраз або n-грам (тригерів), незалежно від їхньої точної позиції в тексті. Властивість трансляційної інваріантності CNN дозволяє успішно ідентифікувати такі ознаки, навіть якщо вони зміщені всередині речення, тоді як RNN може бути чутливою до позиційних змін.

Крім того, CNN демонструють більшу стійкість до шуму в даних. Оскільки операція max-pooling відбирає лише найсильніші ознаки з карти ознак, випадкові або незначущі слова (шум) відфільтровуються і не впливають на фінальне рішення класифікатора. У рекурентних мережах кожне слово, навіть незначуще, проходить через оновлення прихованого стану, що може вносити "сміття" в контекстний вектор. Тому в умовах реальних задач, де тексти часто містять помилки, зайві символи або неструктуровані вставки, згорткові архітектури часто забезпечують кращу узагальнюючу здатність при менших витратах на налаштування гіперпараметрів, рис.1.5.

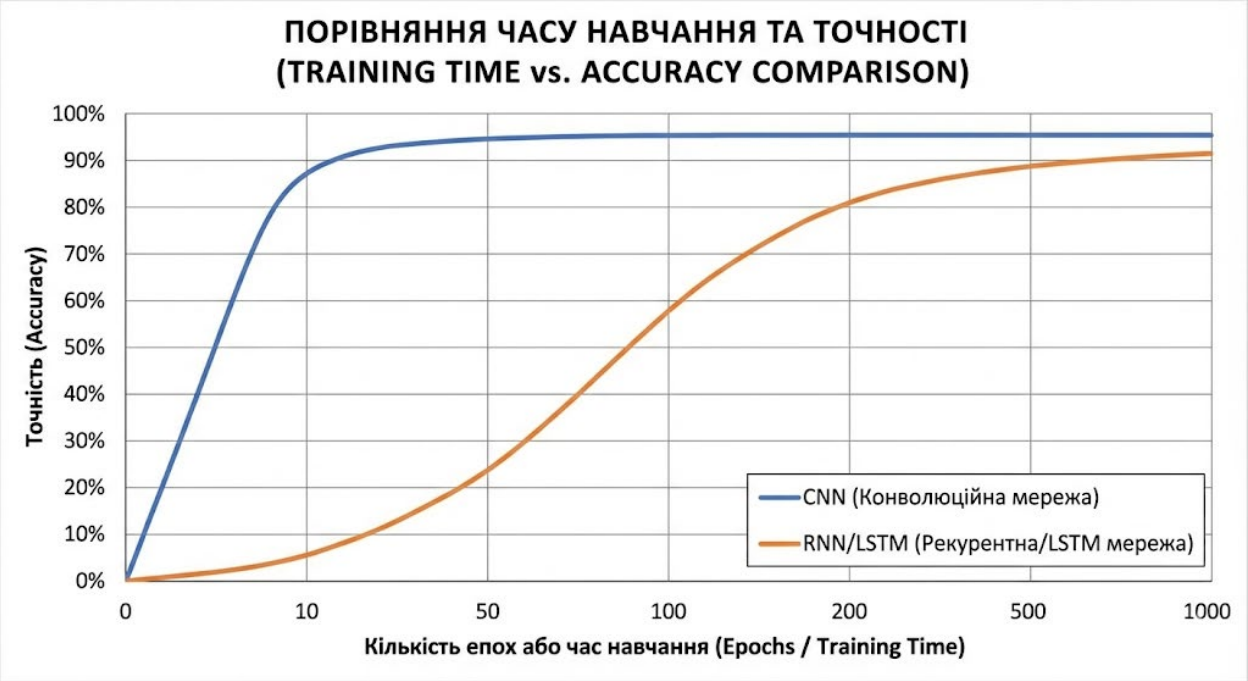



Рисунок 1.5 — Порівняльна характеристика швидкості збіжності алгоритмів навчання CNN та LSTM на задачах класифікації текстових масивів великої розмірності

# 2 АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ПРОГРАМНИХ РІШЕНЬ ТА МЕТОДІВ РОЗПІЗНАВАННЯ

## 2.1 Аналіз функціональних можливостей програмного комплексу Microsoft Math Solver.

Одним із найбільш відомих та технологічно досконалих аналогів у досліджуваній предметній області є програмний продукт, розроблений корпорацією Microsoft, який позиціонується як універсальний інструмент для вирішення математичних задач різного рівня складності. Дана система являє собою яскравий приклад застосування передових алгоритмів штучного інтелекту в освітній сфері. Платформа доступна користувачам як у вигляді веб-сервісу, так і у форматі мобільного додатку, що забезпечує високу гнучкість використання. Основний інтерес у контексті даної магістерської роботи становить реалізована в цьому продукті підсистема комп'ютерного зору, яка відповідає за первинну обробку вхідних даних.

Процес взаємодії користувача з системою може відбуватися кількома шляхами, серед яких ключовим є оптичне розпізнавання символів. Додаток дозволяє використовувати камеру мобільного пристрою для сканування друкованого або рукописного тексту безпосередньо з паперових носіїв чи дошок, рис.2.1-2.3.

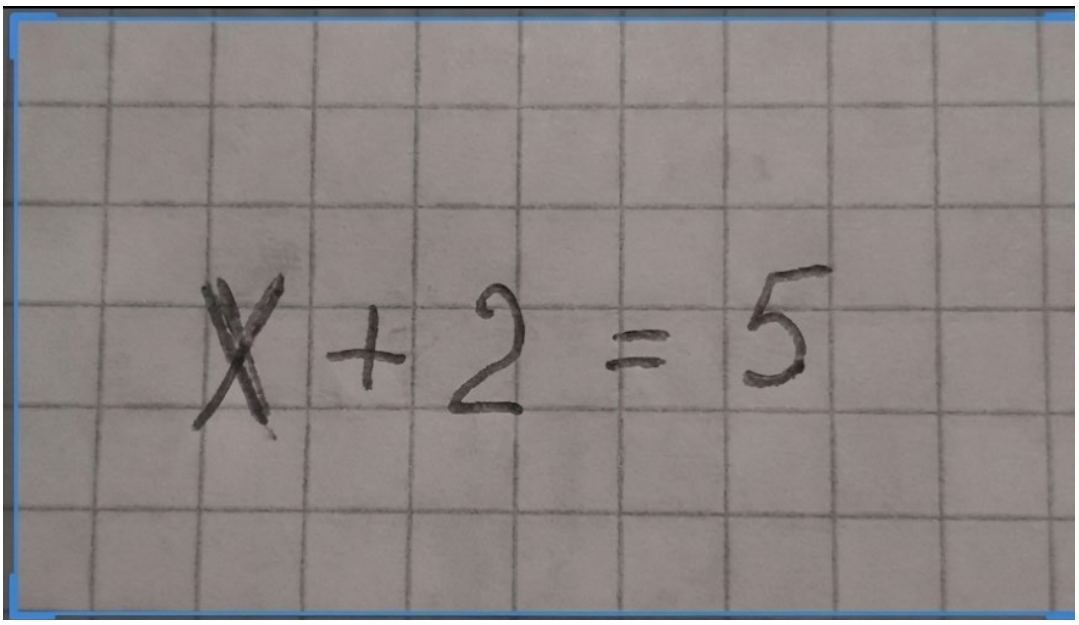



Рисунок 2.1 – Приклад роботи Microsoft Math Solver

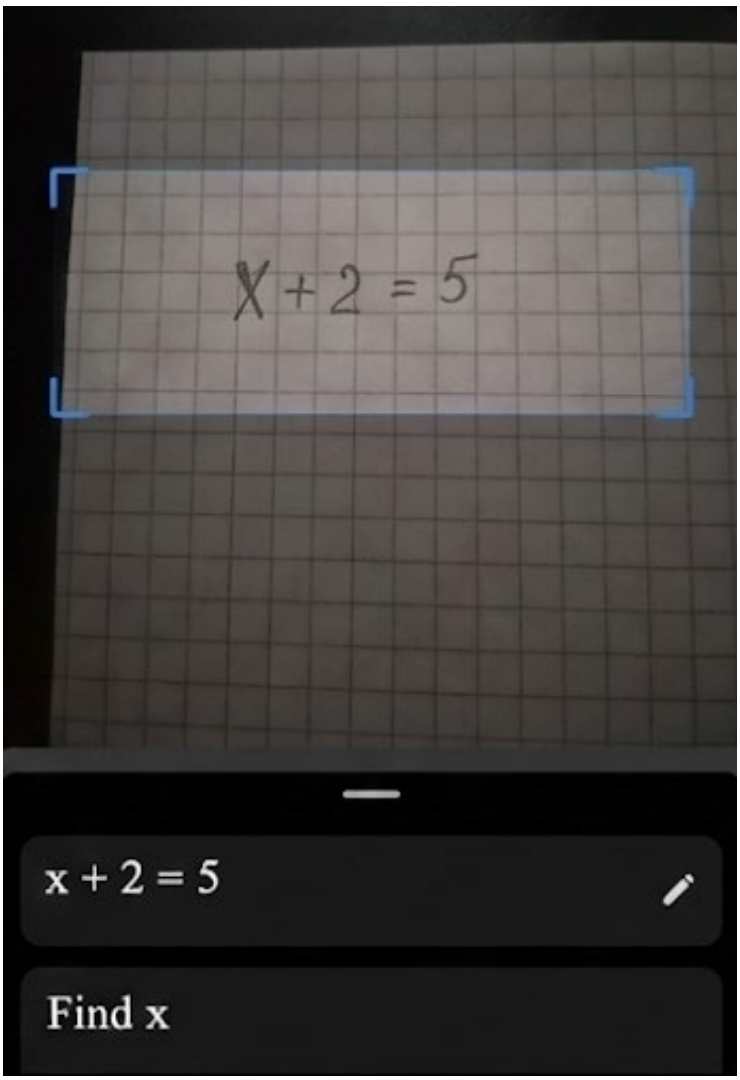



Рисунок 2.2 – Приклад роботи Microsoft Math Solver

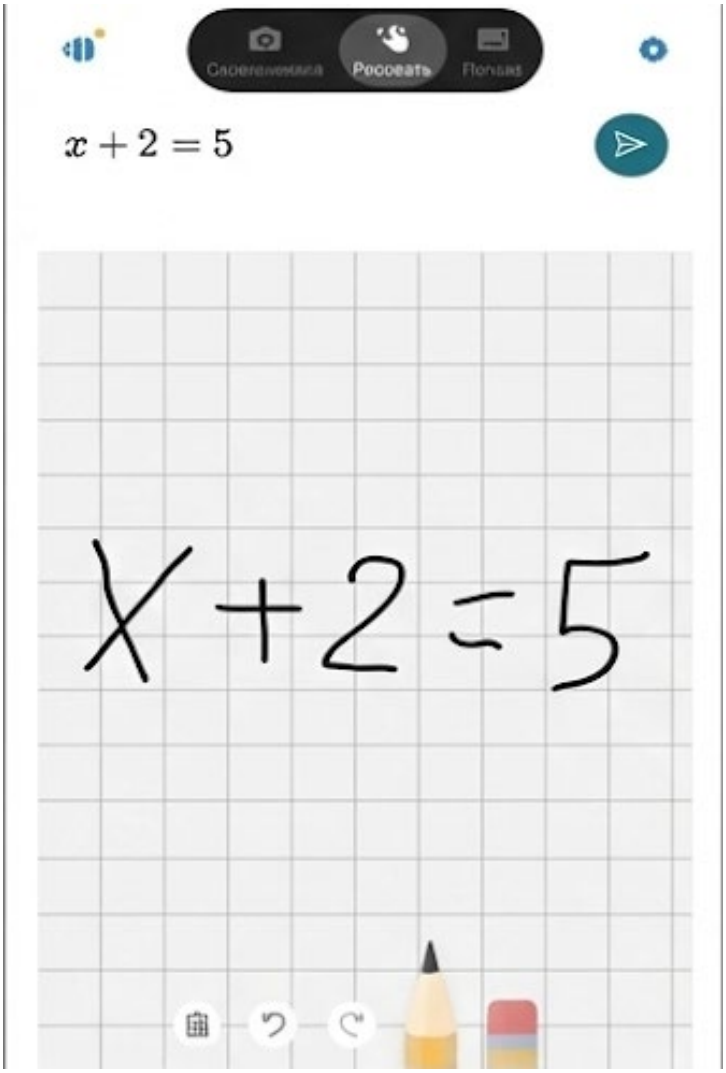



Рисунок 2.3 – Приклад роботи Microsoft Math Solver

Використовувані в ядрі системи нейронні мережі демонструють високу точність у детекції окремих символів та збереженні структурних зв'язків між елементами математичного виразу, навіть за умов нерівномірного освітлення або нечіткого почерку. Окрім фотофіксації, програмне забезпечення підтримує режим малювання на екрані пристрою, що імітує звичайне написання формул на папері та дозволяє системі аналізувати траєкторію руху пальця або стилуса в реальному часі для покращення точності розпізнавання.

Після успішного етапу розпізнавання та перетворення графічного образу в машинний код, система переходить до етапу обчислення. Важливою конкурентною перевагою даного аналога є те, що він не обмежується видачею лише кінцевої відповіді. Програмний комплекс генерує детальні покрокові

пояснення процесу розв'язання, часто пропонуючи кілька різних методів вирішення однієї й тієї ж задачі. Крім того, інструментарій включає потужний модуль графічної візуалізації, який дозволяє будувати графіки функцій у двовимірному та тривимірному просторах, що значно полегшує сприйняття аналітичної інформації користувачем. Такий підхід робить систему не просто калькулятором, а повноцінним навчальним середовищем.

Узагальнюючи аналіз даного аналога, можна стверджувати, що Microsoft Math Solver встановлює високий стандарт якості для систем автоматизованого розпізнавання математичних виразів. Його сильні сторони полягають у поєднанні надійних алгоритмів оптичного розпізнавання символів з потужним математичним ядром та зручним інтерфейсом користувача. Проте, як і будь-яка пропрієтарна система такого масштабу, вона має закритий вихідний код, що унеможливлює детальне вивчення архітектури використовуваних нейронних мереж, але дозволяє виділити ключові вимоги до функціоналу системи, що розробляється в рамках магістерської роботи.

## 2.2 Огляд функціональних можливостей та алгоритмів роботи системи PhotoMath

Ще одним визначним прикладом у сфері автоматизованої обробки математичних даних є програмний продукт PhotoMath, який завоював значну популярність серед користувачів завдяки своїй швидкодії та інтуїтивно зрозумілому інтерфейсу. Ця система по праву вважається одним із піонерів у масовому впровадженні технологій комп'ютерного зору на мобільних платформах для освітніх цілей. Основою функціонування додатку є складна комбінація алгоритмів обробки зображень та глибокого машинного навчання, що дозволяє в реальному часі аналізувати відеопотік з камери смартфона,

виділяти математичні вирази та конвертувати їх у машинно-зчитуваний формат для подальшого обчислення.

Процес розпізнавання у даній системі реалізовано з високим ступенем оптимізації. Користувачеві надається спеціальна рамка видошукача, яку необхідно навести на математичний вираз, рис.2.4.

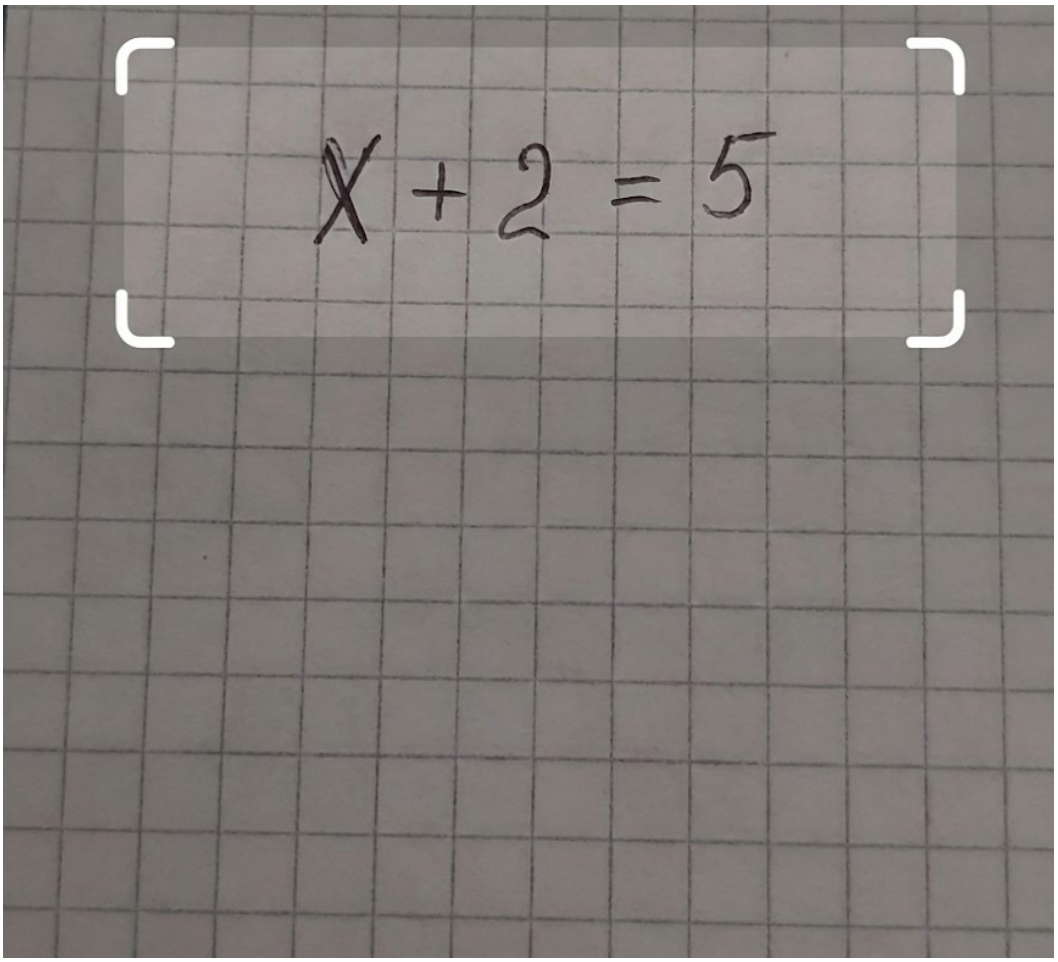



Рисунок 2.4 – Приклад роботи PhotoMath

У цей момент програмне забезпечення виконує попередню обробку зображення, яка включає бінаризацію, видалення шумів та корекцію перспективи, що є критично важливим для коректної роботи нейронної мережі. Система здатна ефективно розпізнавати не лише стандартний друкований текст із підручників, але й рукописні формули, що свідчить про використання великих наборів даних для навчання моделей, які враховують варіативність почерків та різні стилі написання символів.

Після успішної ідентифікації символів та відновлення структури рівняння, дані передаються до обчислювального ядра. Важливою характеристикою PhotoMath є його здатність розкладати складні математичні задачі на прості послідовні кроки. Система не просто видає кінцевий числовий результат, а формує детальну інструкцію з розв'язання, супроводжуючи кожен етап текстовими поясненнями та посиланнями на відповідні математичні правила. Це перетворює процес використання додатку з простого отримання відповіді на процес навчання. Окрім алгебраїчних перетворень, функціонал включає можливість побудови графіків, що дозволяє користувачеві візуально оцінити поведінку функцій та знайти корені рівнянь графічним методом.

Аналізуючи PhotoMath як аналог проектованої системи, варто відзначити високу стійкість його алгоритмів до зовнішніх факторів, таких як недостатнє освітлення або тремтіння камери. Розробники досягли цього завдяки використанню ефективних методів стабілізації та адаптивної корекції зображення. Хоча точна архітектура нейронних мереж, що використовуються в PhotoMath, є закритою інформацією, очевидно, що в її основі лежать принципи згорткових нейронних мереж, подібні до тих, що розглядаються в даній магістерській роботі. Вивчення користувацького досвіду та сценаріїв використання цього додатку дозволяє сформулювати чіткі вимоги до інтерфейсу та швидкодії власної розробки.

## 2.3 Особливості реалізації рукописного введення та розпізнавання формул у середовищі Microsoft OneNote

Окремої уваги в контексті огляду існуючих рішень заслуговує програмний продукт Microsoft OneNote, який, хоч і позиціонується насамперед як цифровий записник для організації інформації, володіє потужним вбудованим інструментарієм для роботи з математичними об'єктами. На

відміну від спеціалізованих мобільних сканерів, розглянутих у попередніх підрозділах, OneNote фокусується на сценарії активного створення контенту, зокрема, через механізми рукописного введення за допомогою стилуса або сенсорного екрана. Цей додаток є частиною ширшої екосистеми офісних програм і демонструє відмінний від інших підхід до взаємодії з користувачем, де процес розпізнавання інтегровано безпосередньо в робочий простір документа. Це дозволяє поєднувати звичайний текст, графічні замальовки та математичні обчислення на одній "нескінченній" сторінці.

Ключовою функцією, що становить інтерес для даного дослідження, є інструмент перетворення рукописного вводу в математичний вираз. Користувач має можливість писати формули від руки безпосередньо на екрані пристрою, імітуючи роботу в звичайному зошиті, рис.2.5-2.6.

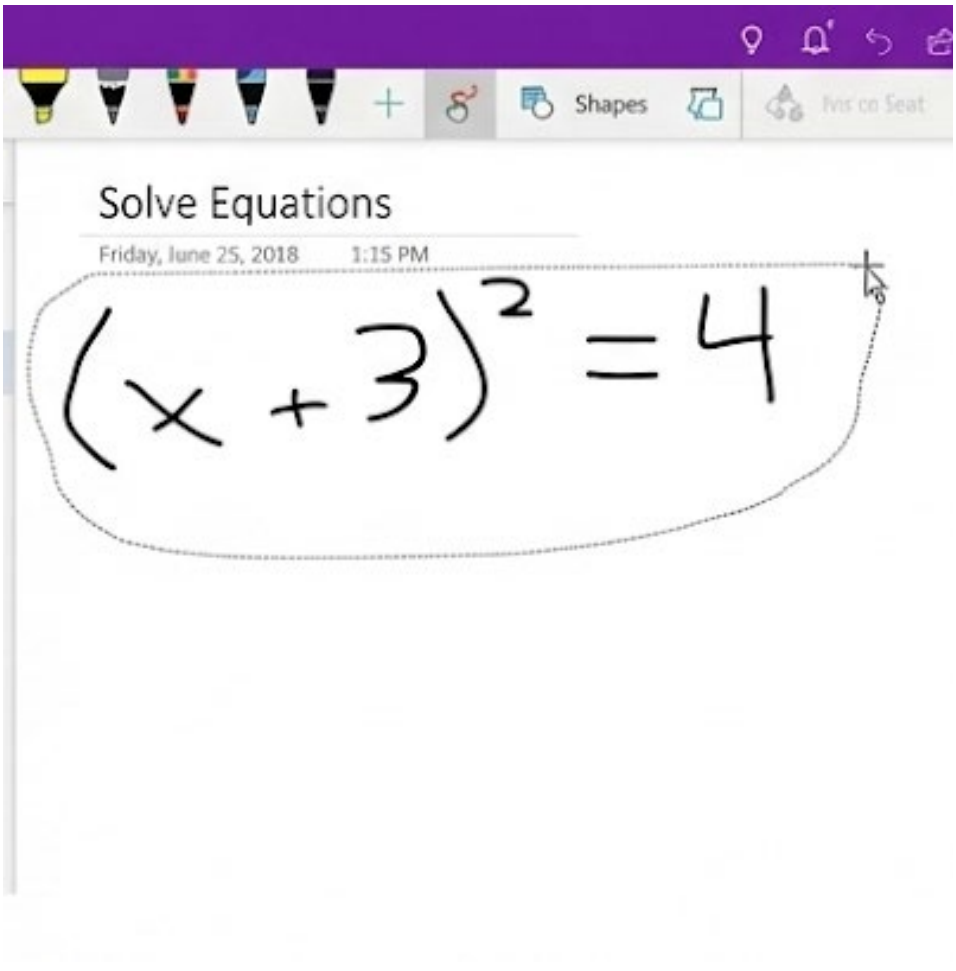



Рисунок 2.5 – Приклад роботи Microsoft OneNote

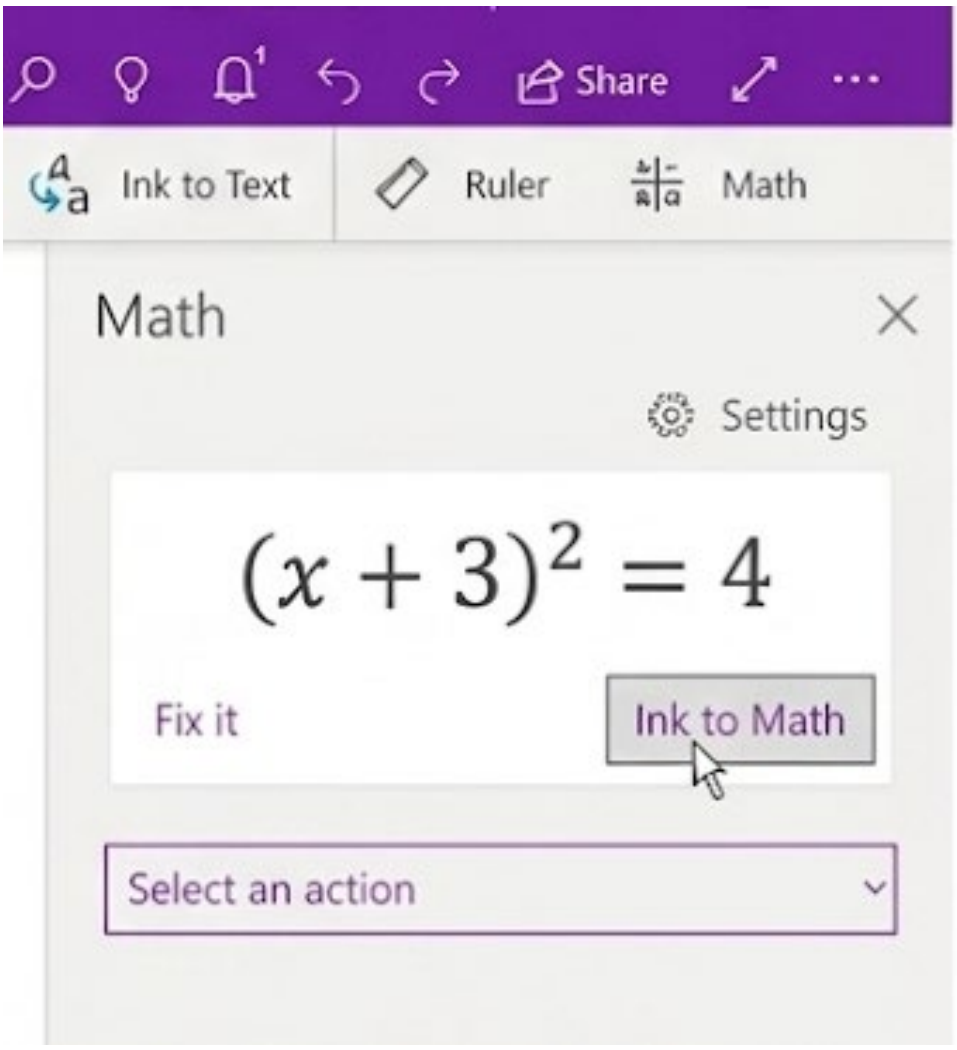



Рисунок 2.6 – Приклад роботи Microsoft OneNote

Після виділення написаного фрагмента спеціальним інструментом "ласо", система активує алгоритми розпізнавання образів, які аналізують геометрію штрихів, їх взаємне розташування та контекст. Важливою архітектурною особливістю тут є використання технологій розпізнавання "онлайн", тобто аналізу траєкторії руху пера в реальному часі, що часто дає вищу точність порівняно зі статичним аналізом зображень, оскільки система має інформацію про порядок написання символів.

Після первинної обробки OneNote відображає розпізнану версію формули у друкованому вигляді поруч із оригінальним рукописним варіантом. Суттєвою перевагою цього аналога є наявність інтерактивного механізму виправлення помилок. Якщо нейронна мережа невірно ідентифікувала певний символ або структурний елемент (наприклад, переплутала індекс зі

множником), користувач може вручну вказати на помилковий фрагмент і вибрати правильний варіант із запропонованого списку альтернатив. Це забезпечує гнучкість системи та дозволяє працювати навіть з недбалим почерку. Після підтвердження коректності розпізнавання, вбудований математичний асистент здатний виконати обчислення, спростити вираз або побудувати графік функції безпосередньо на сторінці нотатки.

Підсумовуючи, можна сказати, що Microsoft OneNote демонструє високу ефективність у задачах сегментації та структурного аналізу складних математичних виразів. Хоча його основний сценарій використання відрізняється від класичних фото-калькуляторів, алгоритми, закладені в його основу, є релевантними для теми магістерської роботи. Вони показують важливість не лише коректної класифікації окремих символів, але й необхідність глибокого розуміння двовимірної структури математичної нотації. Досвід реалізації інтерфейсу виправлення помилок у OneNote може бути корисним при проектуванні модуля взаємодії з користувачем у розроблюваній системі.

# 3 МЕТОДИКА ПРАКТИЧНОЇ РЕАЛІЗАЦІЇ НЕЙРОМЕРЕЖЕВОЇ МОДЕЛІ ТА АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

## 3.1 Обґрунтування вибору та архітектурні особливості нейронної мережі YOLOv5

Для вирішення задачі локалізації та первинної класифікації математичних виразів на вхідних зображеннях у рамках даної магістерської роботи було обрано архітектуру YOLOv5 (You Only Look Once, версія 5). Цей вибір обумовлений необхідністю забезпечення оптимального балансу між високою швидкістю обробки даних та точністю детектування об'єктів, що є критичним параметром для систем, які потенційно можуть працювати в режимі реального часу або на пристроях з обмеженими обчислювальними ресурсами. На відміну від традиційних двостадійних детекторів, які спочатку генерують набір пропозицій регіонів, а потім послідовно класифікують кожен з них, алгоритми сімейства YOLO концептуально розглядають задачу детекції як єдину регресійну проблему. Такий підхід дозволяє прогнозувати координати обмежувальних рамок та відповідні ймовірності класів безпосередньо з пікселів повного зображення за один прохід нейронної мережі, що суттєво пришвидшує процес. П'ята версія цього алгоритму, яка є де-факто стандартом у багатьох прикладних задачах комп'ютерного зору, таких як детектування автомобільного трафіку, розпізнавання дорожніх знаків чи відстеження біологічних об'єктів, демонструє значні покращення порівняно з попередниками завдяки оптимізованій архітектурі та використанню сучасних методів аугментації даних під час навчання.Фундаментальний принцип роботи алгоритму YOLO полягає у специфічному способі обробки вхідного зображення. Система умовно поділяє зображення на сітку розмірністю $S \times S$ комірок. Кожна з цих комірок несе відповідальність за виявлення та

локалізацію об'єкта, якщо центр цього об'єкта геометрично потрапляє в межі даної комірки. Відповідно, кожна комірка сітки повинна передбачити фіксовану кількість обмежувальних рамок, їхні координати відносно меж самої комірки, а також оцінку впевненості (confidence score), яка відображає ймовірність присутності об'єкта в рамці та точність її позиціонування. Окрім того, для кожної рамки прогнозується набір умовних ймовірностей приналежності об'єкта до певного класу. Однак, такий підхід неминуче призводить до генерації надмірної кількості прогнозів, оскільки один і той самий об'єкт може бути "виявлений" декількома сусідніми комірками з дещо відмінними параметрами рамок.Для вирішення проблеми дублювання детекцій та відбору найбільш релевантних результатів детектор використовує алгоритм придушення немаксимумів (Non-Maximum Suppression, NMS). Цей механізм дозволяє ефективно відфільтрувати зайві обмежувальні рамки, залишаючи лише ті, що мають найвищі показники ймовірності. Процес NMS реалізується ітеративно: спочатку всі передбачені рамки сортуються за спаданням їхньої оцінки впевненості. Потім система вибирає рамку з найвищою оцінкою як "переможця" і пригнічує (видаляє) всі інші рамки, які мають значне геометричне перекриття з нею. Ступінь перекриття визначається метрикою "перетин над об'єднанням" (Intersection over Union, IoU). Якщо значення IoU перевищує певний встановлений поріг, рамка вважається дублікатом і відкидається. Цей процес повторюється для всіх класів об'єктів, гарантуючи, що кожен об'єкт буде представлений лише однією, найбільш точною рамкою. Загальна схема роботи алгоритму YOLO, від подачі вхідного зображення до отримання фінальних детекцій після застосування NMS, наочно продемонстрована на рис.3.1.

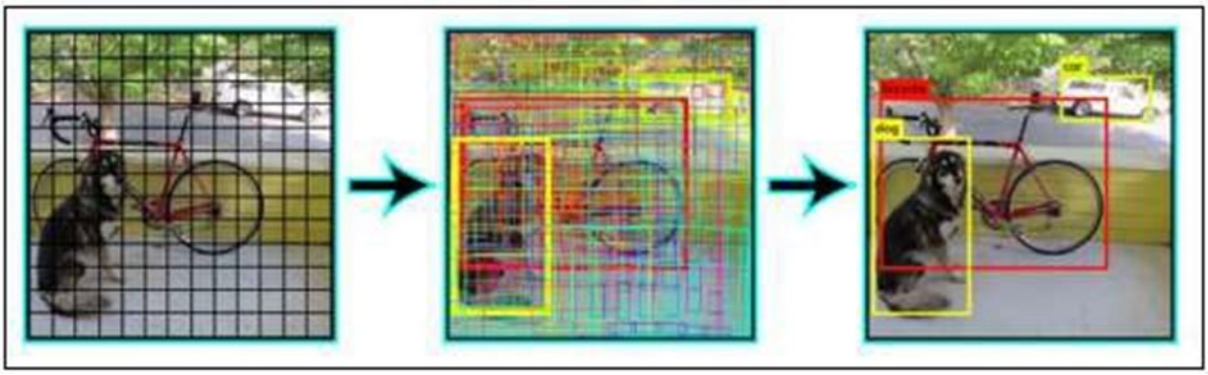


Рисунок 3.1 – Демонстрація принципу роботи алгоритму YOLO

Важливим аспектом практичної реалізації є те, що YOLOv5 є розробкою з відкритим вихідним кодом, що надає дослідникам широкий набір попередньо навчених моделей різної складності та інструментів для їхнього донавчання (fine-tuning) на власних даних. Базові моделі, як правило, проходять тривале попереднє навчання на масштабних загальнодоступних наборах даних, таких як COCO, що дозволяє їм сформувати потужні екстрактори ознак ще до початку роботи зі специфічними математичними даними.

Враховуючи складність структури математичних виразів, для ефективної роботи системи розпізнавання було прийнято рішення розділити загальний процес на два послідовні ієрархічні етапи, кожен з яких обслуговується окремою нейромережевою моделлю. Перший етап полягає у розпізнаванні та локалізації повноцінної математичної формули на зображенні як єдиного цілісного об'єкта. Завданням цього етапу є відокремлення корисної області з формулою від фону та інших сторонніх об'єктів. Другий етап передбачає детальну детекцію кожного окремого математичного символу виключно в межах тієї області, яка була знайдена та вирізана на попередньому кроці. Такий підхід дозволяє значно спростити завдання для другої нейромережі, оскільки вона працює з очищеними даними без фонового шуму, що підвищує загальну точність розпізнавання.

Для реалізації цього двоетапного підходу було підготовлено два окремі, незалежно розмічені набори даних: один для детекції формул, другий — для детекції окремих символів. Варто зазначити, що на поточному етапі дослідження набори даних містять переважно друковані символи. Процес підготовки даних включав збір зображень, їх фільтрацію та анотування. Для набору даних, призначеного для навчання першої моделі (детектора формул), початкова кількість зображень становила 321 одиницю, на яких було розмічено 1642 області з формулами. Паралельно, для набору даних із символами було зібрано 544 зображення, що містили 6664 розмічені області. Обидва набори даних пройшли ретельну процедуру очищення від помилкових розміток, дублікатів та нерелевантних даних, які були віднесені до категорії "other class". Загалом у процесі чистки було видалено 1781 область, що дозволило підвищити якість навчальної вибірки.

Процес анотування (розмітки) графічних даних є критично важливим етапом підготовки до навчання нейронної мережі. Для виконання цього завдання використовувалася спеціалізована хмарна платформа Roboflow, яка надає зручний графічний інтерфейс для виділення об'єктів обмежувальними рамками та присвоєння їм відповідних класів. Приклад робочого середовища платформи Roboflow під час процесу розмітки зображення наведено на рис.3.2.




Рисунок 3.2 – Інтерфейс платформи для розмітки даних Roboflow

Джерела даних для формування навчальних наборів були комбінованими. Частина зображень була зібрана вручну шляхом фотографування друкованих матеріалів та рукописних записів, а інша частина була запозичена з відкритих репозиторіїв платформи Roboflow. Важливо було забезпечити репрезентативність даних, тому до набору були включені зображення різних типів математичних виразів. Наприклад, для навчання детектора формул використовувалися зображення аркушів, що містили лінійні рівняння, створені вручну, як показано на рис.3.3.

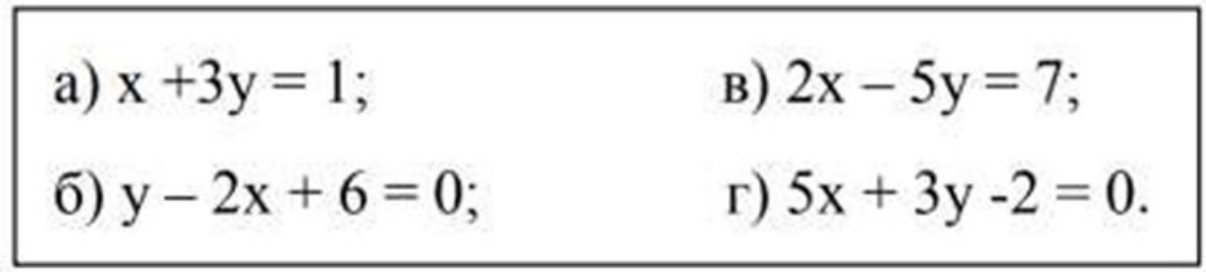



Рисунок 3.3 – Приклад аркуша з рукописними лінійними рівняннями

Крім того, з відкритих джерел були відібрані зображення, що належать до чотирьох різних класів математичних виразів: тригонометричні функції,

границі, інтеграли та диференціальні рівняння. Всі запозичені зображення пройшли ручну верифікацію на предмет якості самого зображення та достовірності наявної розмітки. Типовий приклад зображення, що містить інтегральний вираз і було включено до навчального набору, представлено на рис.3.4.

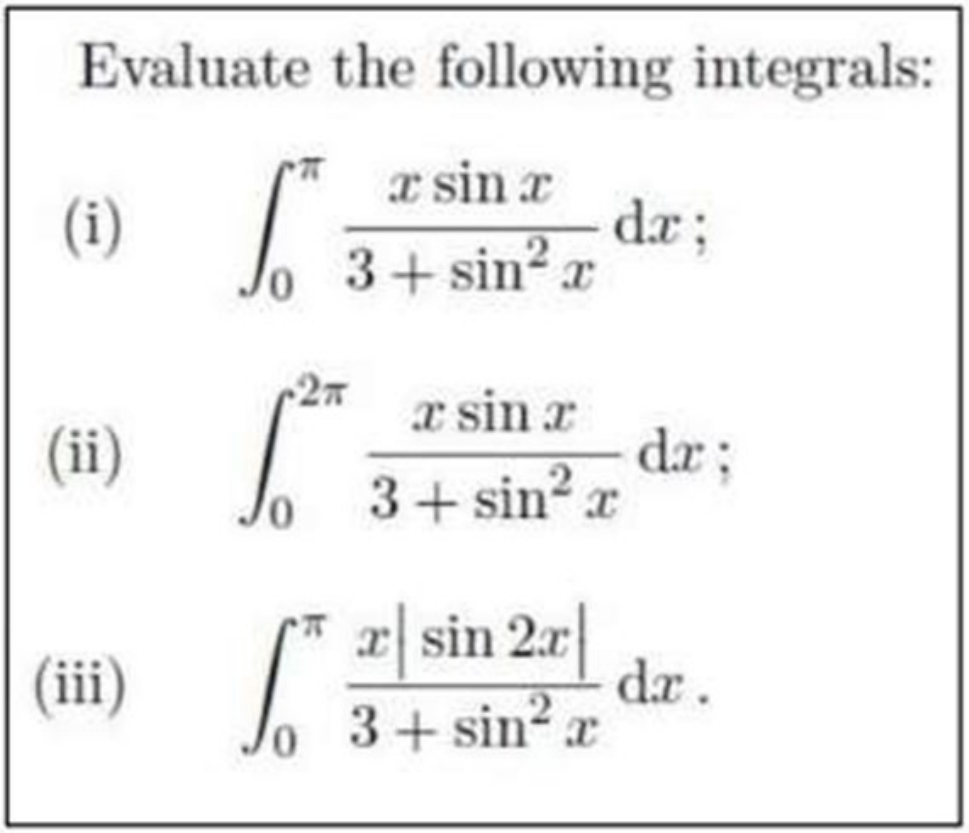



Рисунок 3.4 –Інтегральні вирази

Після всіх етапів збору, фільтрації та об'єднання, підсумковий навчальний набір даних для першої моделі (детектора формул) складався з 721 зображення, що охоплювали вирази з тригонометрією, диференціалами, інтегралами, границями та лінійними рівняннями.

Аналогічний підхід застосовувався і для формування другого набору даних, призначеного для розпізнавання окремих символів. З відкритих джерел готових наборів даних були відібрані зображення, що містять різні класи елементарних математичних об'єктів. До переліку класів увійшли арабські цифри від 0 до 9, найбільш вживані змінні латинського алфавіту, позначення

основних тригонометричних функцій (синус, косинус, тангенс, котангенс), а також базові математичні оператори (додавання, віднімання, множення, ділення, знак рівності). Приклад зображення з вихідного набору даних, що демонструє розмітку окремих символів усередині складнішого виразу, показано на рис.3.5.

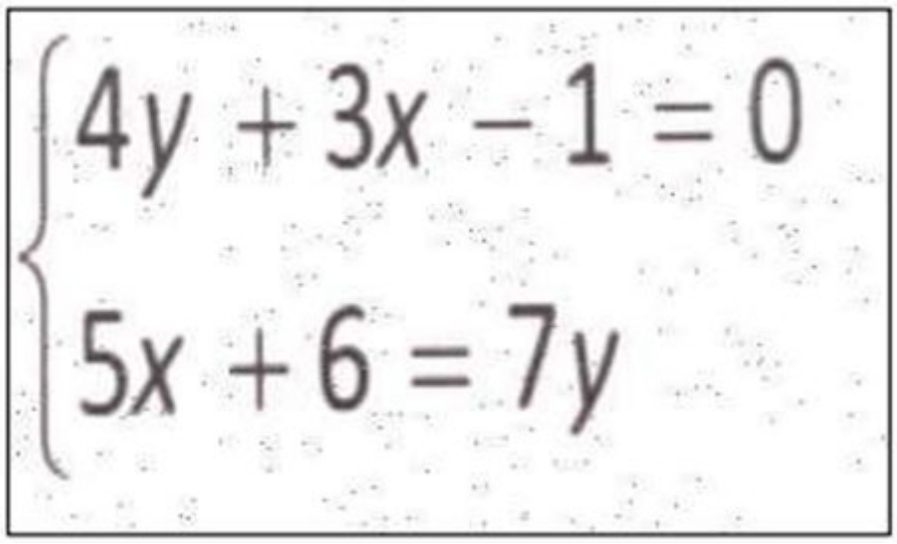



Рисунок 3.5 –Зображення з набору даних із розміченими символами

Для підвищення стійкості моделі до варіацій у написанні символів, набір даних був додатково розширений спеціально створеними зображеннями, що містили друковані символи різних шрифтів та розмірів. Приклад такої додаткової розмітки друкованих символів представлено на рис.3.6. У результаті загальна кількість зображень у фінальному наборі даних, призначеному для навчання нейронної мережі розпізнаванню окремих символів, склала 899 одиниць.

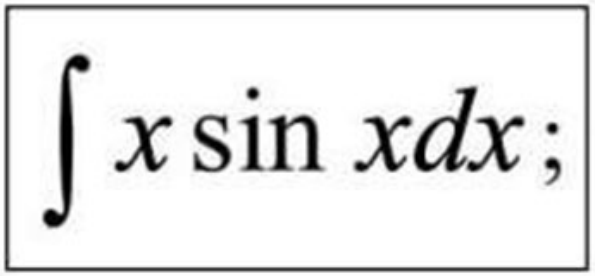



Рисунок 3.6 – Додаткова розмітка друкованих символів

Перед початком програмної реалізації системи були чітко сформульовані вимоги до неї, які поділяються на функціональні та нефункціональні. Функціональні вимоги описують поведінку системи та операції, які вона повинна виконувати. До них відносяться: надання користувачеві можливості завантажити зображення для обробки; обов'язкове виведення системою результату розпізнавання у вигляді текстового повідомлення, що містить відповідну математичну формулу; а також реалізація механізму валідації вхідних файлів, який дозволяє приймати до обробки виключно зображення та відхиляти файли інших форматів.

Нефункціональні вимоги, у свою чергу, визначають якісні атрибути системи та технологічні обмеження. У рамках даного проєкту було визначено, що програмна реалізація має бути виконана з використанням мови програмування Python версії 3.9, яка є стандартом у галузі машинного навчання. Ключовою архітектурною вимогою є використання саме технологій штучних нейронних мереж для виконання основного завдання розпізнавання математичних формул, що обґрунтовує вибір YOLOv5.

Виходячи з поставлених вимог та обраного двоетапного підходу до розпізнавання, була розроблена модульна архітектура програмного застосунку. Система складається з трьох основних компонентів, кожен з яких має свою зону відповідальності: головний модуль (Main), модуль детекції формул (DetectFormula) та модуль детекції символів (DetectSymb). Взаємодія між цими компонентами та загальна структура системи відображені на діаграмі компонентів, представленій на рис.3.7.

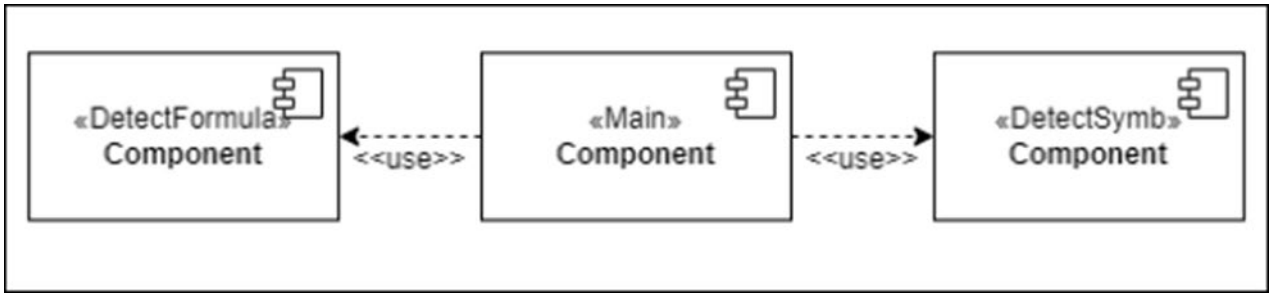



Рисунок 3.7 – Діаграма компонентів програмної системи

Модуль "Main" виступає в ролі центрального контролера застосунку. Він ініціює взаємодію з користувачем, зокрема, через реалізований інтерфейс телеграм-бота, та управляє роботою системи в цілому. У цьому модулі зосереджена основна логіка оркестрації процесу обробки зображення, підключення та ініціалізації нейромережевих моделей, а також обробка можливих виняткових ситуацій та помилок. Крім того, тут реалізована важлива функція трансляції внутрішніх умовних позначень розпізнаних символів у загальноприйнятий математичний синтаксис для формування кінцевого результату.

Компонент "DetectFormula" відповідає за реалізацію першого етапу розпізнавання. Він використовує навчену модель YOLOv5 для аналізу повного вхідного зображення та виявлення на ньому областей, що містять цілісні математичні формули. Важливою функціональною особливістю цього модуля є те, що після детекції він не просто повертає координати рамок, а й автоматично виконує операцію "вирізання" (cropping) виявлених фрагментів зображення. Ці фрагменти зберігаються в окрему директорію з проміжними результатами та слугують вхідними даними для наступного етапу обробки.

Завершальним елементом ланцюжка обробки є компонент "DetectSymb". Він отримує на вхід фрагменти зображень, підготовлені попереднім модулем, і використовує другу спеціалізовану нейронну мережу для детекції окремих символів усередині цих фрагментів. Оскільки результат роботи нейромережі є набором невпорядкованих детекцій, критично важливою частиною цього модуля є набір додаткових алгоритмів пост-обробки. Ці функції відповідають за просторове впорядкування виявлених символів, щоб відновити коректну структуру формули зліва направо. Також реалізовано механізми очищення результатів від артефактів, таких як подвійне виявлення одного й того ж символу (що є доповненням до NMS) або видалення помилково детектованих знаків пунктуації, наприклад, крапки з комою в кінці виразу. Окремий алгоритм відповідає за аналіз вертикального положення символів для коректного визначення ступенів та індексів у математичному виразі.

## 3.2 Методика підготовки та анотування графічних даних для навчання нейронних мереж

Критично важливим етапом у процесі розробки системи розпізнавання на базі згорткових нейронних мереж є підготовка якісного навчального набору даних, оскільки точність роботи моделі безпосередньо корелює з коректністю розмітки вхідних зображень. Процес анотування набору даних, що містить друковані математичні символи, здійснювався вручну з використанням спеціалізованого програмного забезпечення. Ця процедура передбачала виділення кожного окремого символу на зображенні за допомогою прямокутної рамки (bounding box), яка має максимально точно охоплювати межі об'єкта, не захоплюючи при цьому зайвого фону або частин сусідніх символів. Після геометричного виділення об'єкта кожній рамці присвоювався відповідний клас, що відповідає семантичному значенню символу (цифра, змінна, оператор тощо). Візуалізацію процесу розмітки окремих символів, де для кожного елемента визначено його межі та клас, наведено на рис.3.8.




Рисунок 3.8 – Приклад розмітки символів у навчальній вибірці

Аналогічний методологічний підхід застосовувався і для формування набору даних, призначеного для розпізнавання цілісних математичних формул. Основна відмінність полягала в масштабі об'єктів: у цьому випадку обмежувальна рамка повинна була охоплювати весь математичний вираз повністю, включаючи всі його складові частини, такі як дроби, інтеграли, ступені та індекси. Це необхідно для того, щоб навчити нейронну мережу першого рівня відокремлювати корисну інформацію від загального фону зображення. Приклад коректно виконаної розмітки для рукописних та друкованих формул продемонстровано на рис.3.9.




Рисунок 3.9 – Приклад розмітки областей з формулами

Важливим аспектом аналізу якості підготовлених даних є оцінка збалансованості класів, оскільки дисбаланс може призвести до того, що модель буде "перенавчена" на розпізнавання більш частотних об'єктів і ігноруватиме рідкісні. Детальну статистичну характеристику розподілу класів у сформованому наборі даних формул візуалізовано на діаграмі, представленій на рис.3.10. Цей графік дозволяє оцінити репрезентативність вибірки та за необхідності скоригувати набір даних шляхом додавання прикладів для менш представлених класів.




Рисунок 3.10 – Діаграма балансу кількості розмічених областей для формул

У ході роботи з даними, запозиченими з відкритих джерел, було виявлено низку суттєвих недоліків у їх первинній розмітці. Велика кількість зображень містила помилкові анотації, де клас символу не відповідав його графічному зображенню, або межі рамки були встановлені неточно. Для забезпечення високої точності майбутньої системи всі зовнішні набори даних пройшли етап ретельної ручної верифікації та корекції. Типовим прикладом такої помилки є ситуація, коли цифра "6" була помилково промаркована як цифра "3" або інший схожий за накресленням символ. Приклад виявленої некоректної розмітки, яка могла б негативно вплинути на процес навчання, показано на рис.3.11.

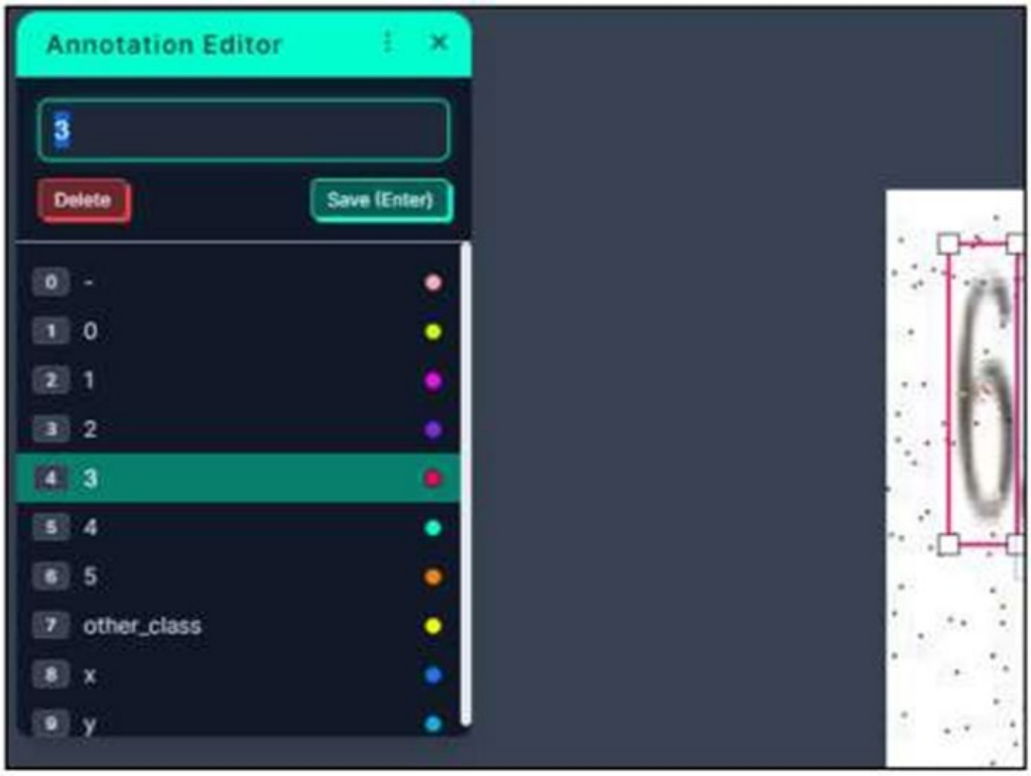



Рисунок 3.11 – Приклад виявленої помилки розмітки (цифра 6 помилково класифікована як 3)

Після виявлення подібних артефактів проводилася процедура виправлення анотацій: хибні мітки замінювалися на правильні, а геометрія рамок коригувалася відповідно до реальних контурів символів. Результат виправлення розмітки для того ж самого фрагмента зображення наведено на рис.3.12. Така скрупульозна перевірка дозволила очистити навчальну вибірку від шумів та підвищити надійність алгоритму розпізнавання.

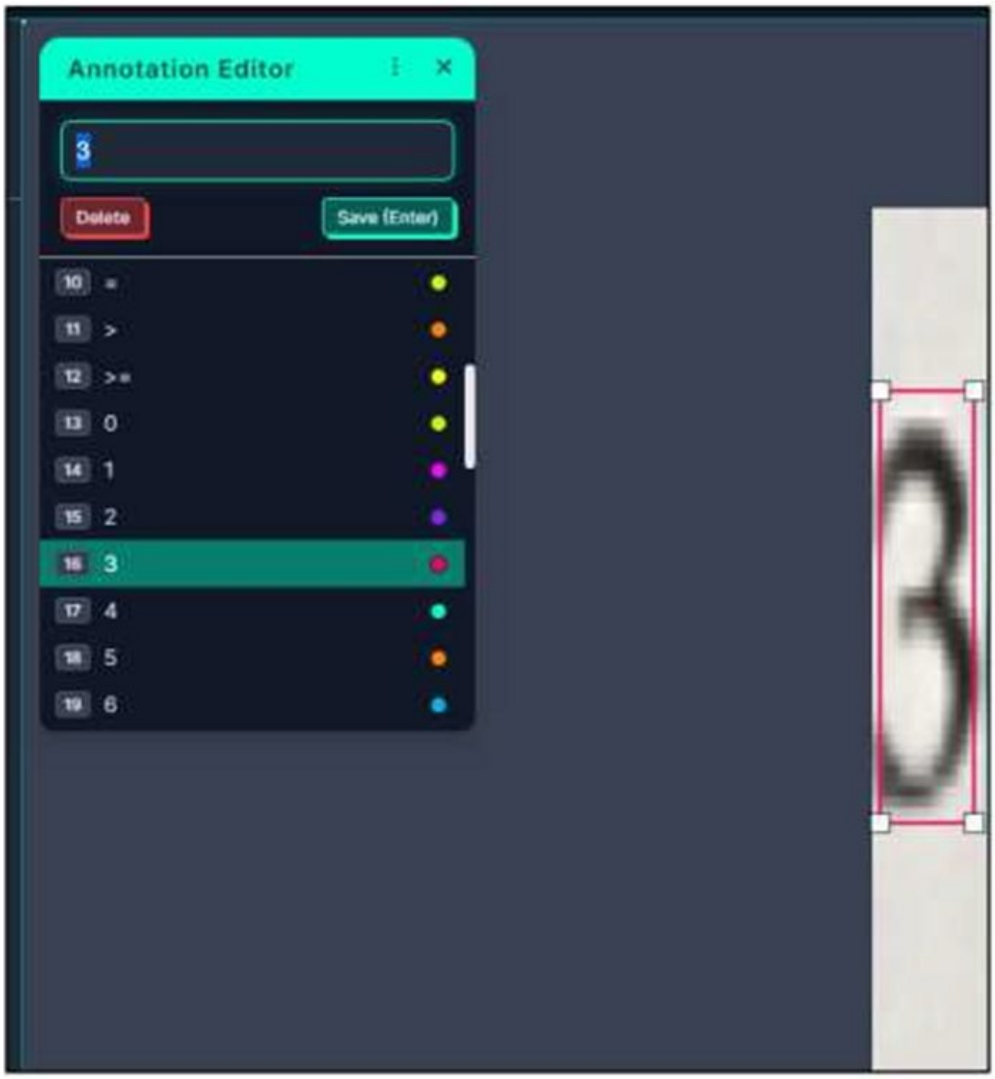



Рисунок 3.12 – Приклад виправленої та верифікованої розмітки

За аналогією з набором формул, для набору даних окремих символів також було проведено аналіз розподілу кількості прикладів по кожному класу. Це дозволило переконатися, що нейронна мережа отримає достатню кількість прикладів для вивчення кожного типу символів, від цифр до спеціальних математичних операторів. Графічне відображення балансу класів для друкованих символів представлено на рис.3.13.




Рисунок 3.13 – Баланс кількості розмічених областей для набору друкованих символів

Процес підготовки даних виявився найбільш трудомістким етапом дослідження. Сумарні часові витрати на збір, первинну розмітку, верифікацію та корекцію всіх наборів даних склали приблизно 250 годин. Така значна тривалість обумовлена ітеративним характером роботи: після проведення пробних запусків навчання та аналізу помилок детектування, набори даних неодноразово перезбиралися та доповнювалися новими зображеннями для усунення слабких місць моделі. Варто також зазначити, що майже всі зображення у вибірці представлені у чорно-білому форматі, а на кожному окремому зображенні зазвичай розміщено кілька формул або велика кількість символів, що вимагало високої концентрації уваги при розмітці.

## 3.3 Структура набору даних та методика навчання нейронної мережі

Після завершення етапу анотації сформований набір даних експортується з платформи розмітки для подальшого використання у процесі навчання моделі. Структура вивантаженого пакету даних відповідає стандартним вимогам архітектури YOLO і складається з трьох основних компонентів:

1. Директорія із зображеннями: містить вихідні графічні файли, що подаються на вхід нейронної мережі.

2. Директорія з файлами розмітки: містить текстові файли (формат .txt), назви яких ідентичні відповідним зображенням.

3. Файл конфігурації: визначає шляхи до навчальної та валідаційної вибірок, а також кількість та назви класів.

Кожен файл розмітки містить інформацію про об'єкти, присутні на відповідному зображенні. Кожен рядок файлу описує один об'єкт у форматі: <class_id> <x_center> <y_center> <width> <height> де перша позиція — це індекс класу згідно із загальним списком, а наступні значення — нормалізовані координати центру та габарити обмежувальної рамки (bounding box). Приклад вмісту файлу з параметрами навчання наведено на рис. 3.14.

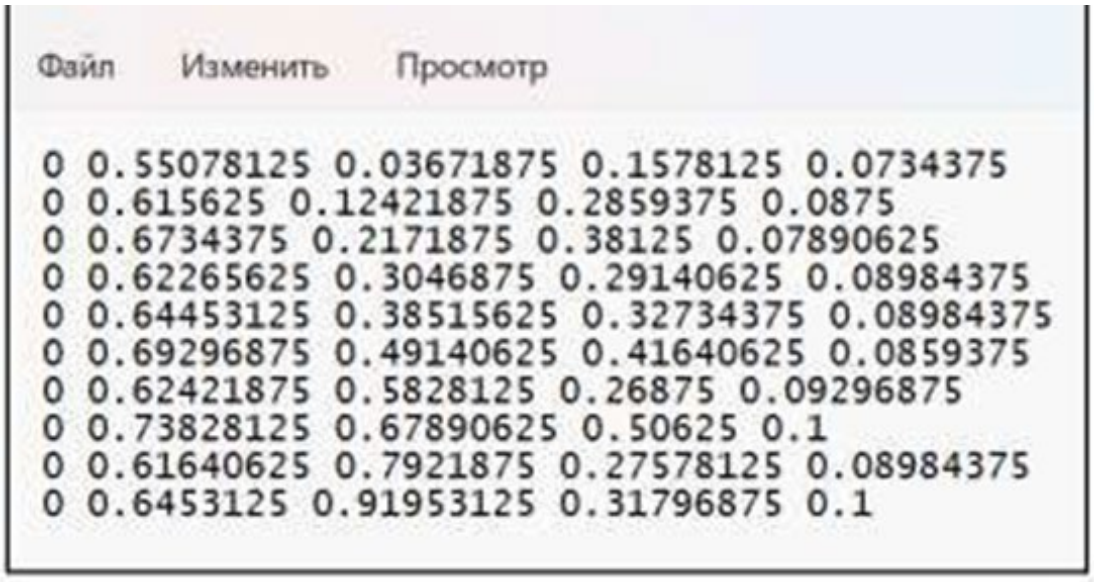
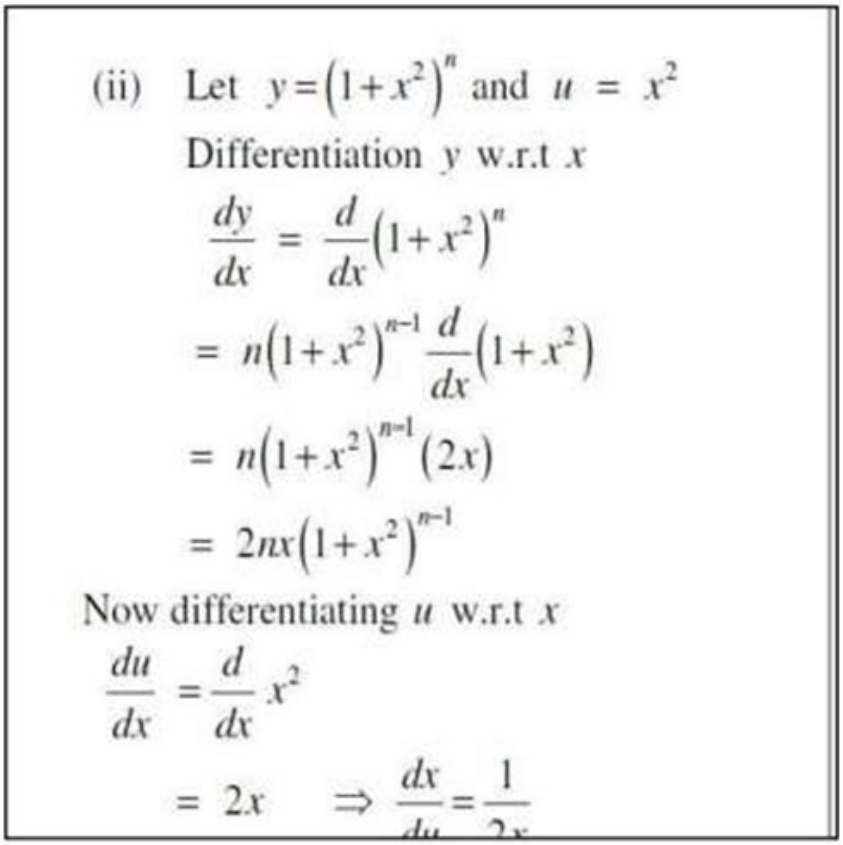



Рисунок 3.14 – Приклад структури текстового файлу розмітки даних

А візуалізацію відповідного розміченого зображення — на рис. 3.15.



(ii) Let $y=\left(1+x^2\right)^n$ and $u = x^2$

Differentiation $y$ w.r.t $x$

$$\frac{dy}{dx} = \frac{d}{dx}\left(1+x^2\right)^n$$

$$= n\left(1+x^2\right)^{n-1}\frac{d}{dx}\left(1+x^2\right)$$

$$= n\left(1+x^2\right)^{n-1}(2x)$$

$$= 2nx\left(1+x^2\right)^{n-1}$$

Now differentiating $u$ w.r.t $x$

$$\frac{du}{dx} = \frac{d}{dx}x^2$$

$$= 2x \quad \Rightarrow \frac{dx}{du} = \frac{1}{2x}$$

Рисунок 3.15 – Візуалізація розміченого зображення з навчального набору

Для реалізації підсистеми детектування було обрано архітектуру YOLOv5s (You Only Look Once, version 5 small). Цей вибір ґрунтується на порівняльному аналізі, проведеному в дослідженні [37], де розглядалися різні модифікації YOLO (s, m, l, x).

Версія YOLOv5s продемонструвала оптимальний баланс для поставленої задачі: вона забезпечує найвищу швидкість обробки кадрів (FPS) та має найменші вимоги до обчислювальних ресурсів і пам'яті GPU серед усіх варіацій 5-го покоління. Це є критичним фактором для систем, що потребують роботи в режимі, наближеному до реального часу. Порівняльний графік швидкості навчання та продуктивності моделей наведено на рис. 3.16.

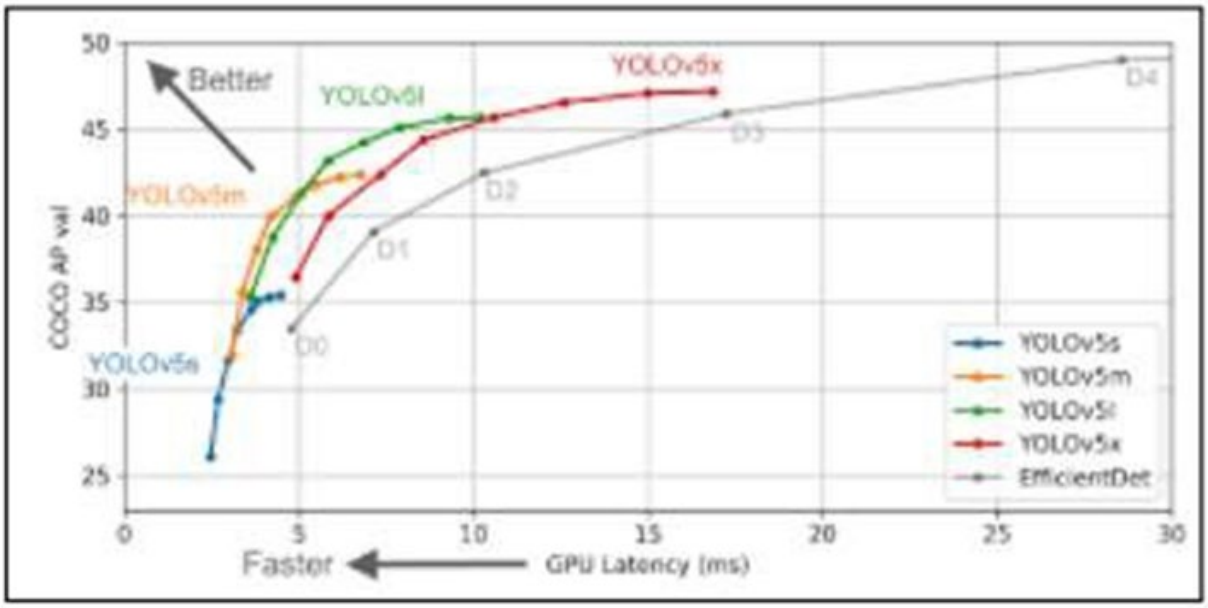



Рисунок 3.16 – Порівняльна діаграма швидкості навчання та продуктивності моделей сімейства YOLO

Архітектура мережі описується у спеціальному конфігураційному файлі формату YAML, який визначає структуру шарів Backbone (виділення ознак), Neck (агрегація ознак) та Head (детектування). Фрагмент програмного коду, що описує конфігурацію моделі, представлений на рис. 3.17.

```
# Parameters
nc: 80  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 v6.0 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Conv, [64, 6, 2, 2]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
   [-1, 3, C3, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8
   [-1, 6, C3, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16
   [-1, 9, C3, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32
   [-1, 3, C3, [1024]],
```

Рисунок 3.17 – Фрагмент конфігураційного YAML-файлу з описом архітектури моделі

З огляду на специфіку предметної області, процес навчання було розділено на дві незалежні ітерації для вирішення двох різних підзадач:

1. Детектування та класифікація математичних формул (як цілісних об'єктів).

2. Детектування та класифікація окремих символів (всередині формул або тексту).

Такий підхід вимагав тренування двох екземплярів нейронної мережі з ідентичною топологією (YOLOv5s), але на різних наборах даних.

Навчання проводилося з використанням хмарного середовища Google Colaboratory, що дозволило залучити необхідні GPU-потужності. У процесі налаштування гіперпараметрів було обрано наступну конфігурацію:

— Оптимізатор: Stochastic Gradient Descent (SGD) — забезпечує стабільну збіжність моделі.

— Функції активації: SiLU (Sigmoid Linear Unit) для прихованих шарів та Sigmoid для вихідного шару.

— Функція втрат (Loss Function): Focal Loss — дозволяє ефективно вирішувати проблему дисбалансу класів, фокусуючись на складних для розпізнавання об'єктах.

Для підвищення узагальнюючої здатності мережі та запобігання перенавчанню до обох наборів даних було застосовано методи аугментації. Штучне розширення вибірки відбувалося за рахунок випадкових геометричних трансформацій (зміна кута нахилу) та зміни фотометричних параметрів (яскравість, контраст, шум).

Навчання моделі для детектування формул Набір даних для цієї задачі складався з приблизно 1300 зображень, що містять формули різної складності. Обсяг навчальної вибірки склав 84% від загального масиву даних, решта була використана для валідації. Експериментальним шляхом було встановлено, що оптимальна тривалість навчання становить 50 епох. Подальше навчання не давало суттєвого приросту точності. Отримані метрики якості:

— mAP@0.5 (Mean Average Precision): 0,983 — вказує на виключно високу точність локалізації об'єктів.

— Precision (Точність): 0,961.

— Recall (Повнота): 0,950.

Графік зміни метрик у процесі навчання, рис. 3.18, демонструє швидку збіжність моделі та стабільність на валідаційній вибірці.

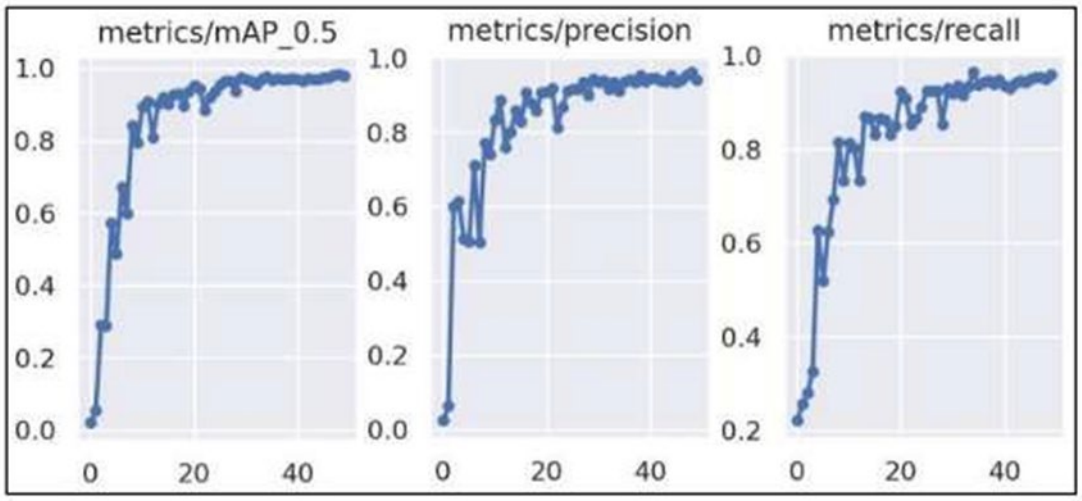



Рисунок 3.18 – Динаміка зміни метрик якості під час навчання моделі детектування формул

Навчання моделі для детектування символів Для задачі розпізнавання окремих символів (таких як тригонометричні функції sin, cos, tg, логарифми ln та змінні) було сформовано більший набір даних — приблизно 2200 зображень. Це зумовлено значно більшою варіативністю класів та меншим розміром самих об'єктів. На навчальну вибірку було відведено 93% даних. З огляду на складність задачі (дрібні об'єкти), навчання тривало довше — 100 епох. Отримані метрики:

— mAP@0.5: 0,898.

— Precision: 0,905.

— Recall: 0,843.

Дещо нижчі показники порівняно з детектуванням цілих формул пояснюються візуальною схожістю деяких символів та складністю їх

виділення у рукописному або зашумленому тексті. Динаміку навчання відображено на рис. 3.19.

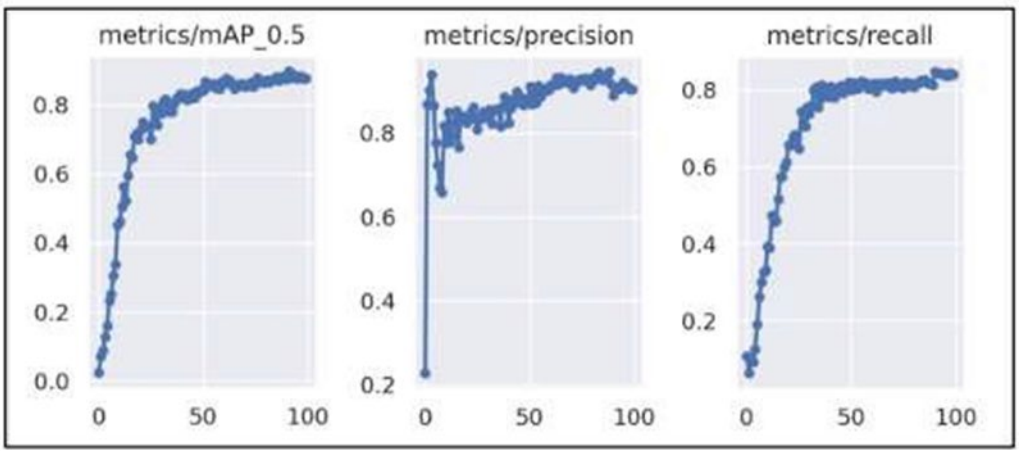



Рисунок 3.19 – Динаміка зміни метрик якості під час навчання моделі розпізнавання символів

## 3.4 Тестування та верифікація роботи нейронної мережі

У даному підрозділі представлено результати експериментальної перевірки працездатності розроблених нейронних мереж. Тестування проводилося окремо для двох навчених моделей: детектора математичних формул (як цілісних структурних одиниць) та детектора окремих символів.

За результатами навчання та валідації на тестовій вибірці було досягнуто високих показників точності, які оцінювалися за метрикою mAP (Mean Average Precision). Ця метрика демонструє середню точність детектування для всіх класів набору даних та є ключовим індикатором якості роботи моделі:

— Для моделі розпізнавання формул показник mAP склав 0,983, що свідчить про майже безпомилкову локалізацію математичних виразів на зображенні.

— Для моделі розпізнавання символів показник mAP склав 0,898. Цей результат є прийнятним, враховуючи складність розділення дрібних та візуально схожих рукописних символів.

Система була навчена розпізнавати об'єкти наступних категорій:

1. Категорія «Формули» (5 класів): інтеграли, системи лінійних рівнянь, диференціальні рівняння, тригонометричні вирази, границі (lim).

2. Категорія «Символи» (23 класи):

— Цифри: 0–9.

— Математичні оператори та знаки: ділення (/), множення (*), додавання (+), віднімання (-), рівність (=), знак інтеграла, дужки.

— Змінні: x, y, z, t, d.

Результатом роботи алгоритму є накладання на вхідне зображення обмежувальних рамок (bounding boxes), які локалізують знайдені об'єкти. Кожна рамка супроводжується текстовою міткою, що містить:

1. Назву передбаченого класу (наприклад, "integral" або "x").

2. Числове значення впевненості (confidence score). Хоча в загальному описі результатів використовується метрика mAP, на конкретному зображенні це число (від 0 до 1) показує ймовірність того, що виділений об'єкт належить до вказаного класу.

Приклад візуалізації детектування з відповідною розміткою наведено на рис. 3.20.

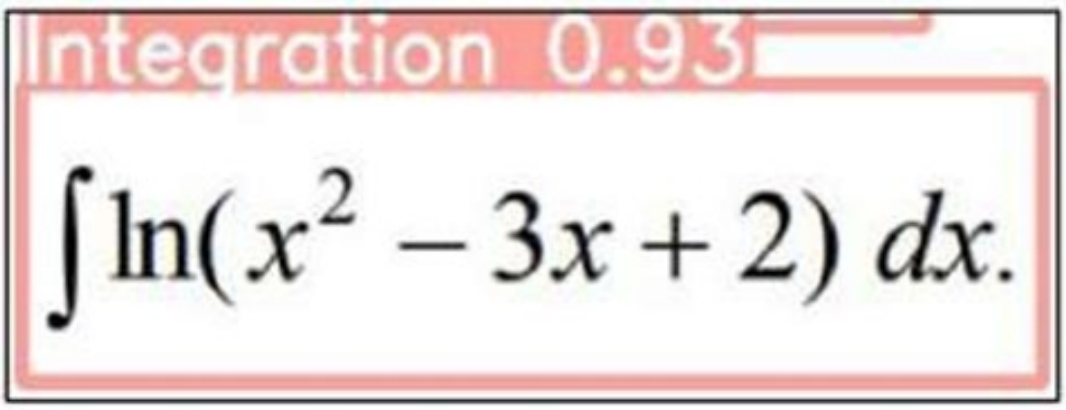



Рисунок 3.20 – Структура візуалізації результату: обмежувальна рамка, назва класу та ступінь впевненості

Першим етапом роботи системи є виділення загальної області формули. Це дозволяє відокремити математичний вираз від фону або іншого тексту. На рис. 3.21 продемонстровано результат успішного детектування рукописної формули, де модель коректно окреслила межі виразу з високим ступенем впевненості.

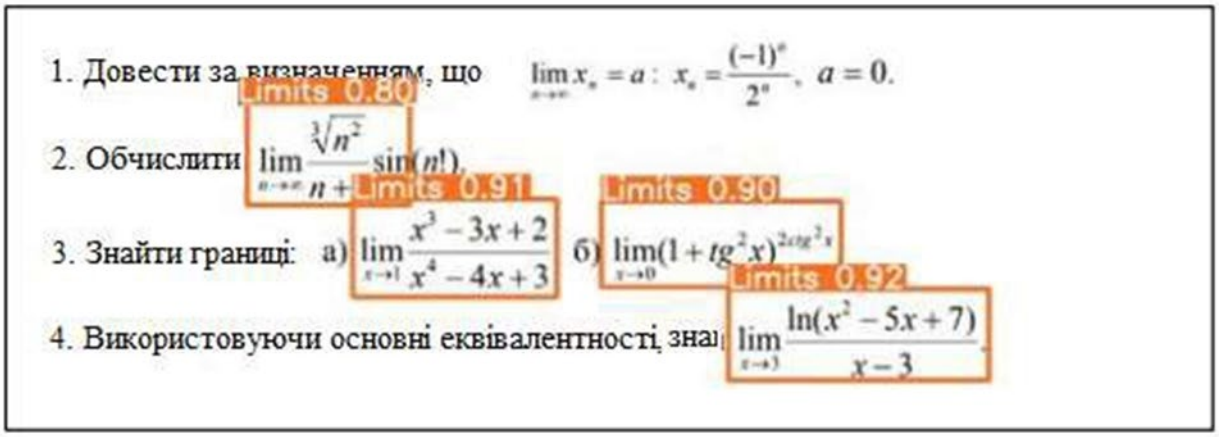



Рисунок 3.21 – Результат роботи нейронної мережі: детектування цілісної формули

Другий етап є більш деталізованим і полягає у сегментації внутрішньої структури формули на окремі складові. Це завдання є складнішим через щільне розташування символів та варіативність почерку. На рис. 3.22 наведено фрагмент вхідного зображення, яке подається на вхід другої нейронної мережі.

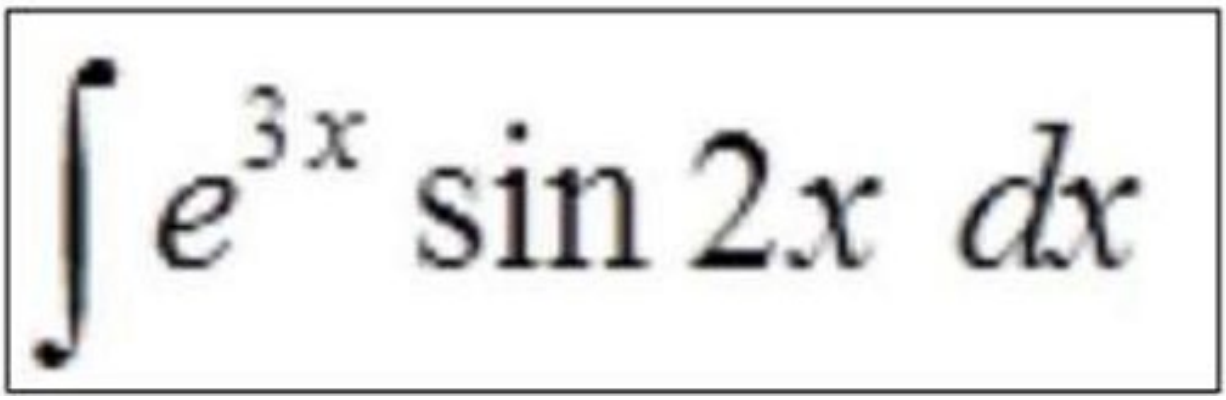



Рисунок 3.22 – Фрагмент вихідного зображення з рукописною формулою

Результат обробки (інференсу) моделі показано на рис. 3.23, де видно, що система успішно ідентифікувала та класифікувала кожен окремий елемент виразу, включаючи змінні, знаки операцій та дужки.




Рисунок 3.23 – Результат посимвольного детектування та класифікації об’єктів

# ВИСНОВКИ

У магістерській кваліфікаційній роботі вирішено актуальне науково-прикладне завдання щодо створення системи розпізнавання математичних формул на основі сучасних методів комп'ютерного зору.

Основні результати роботи полягають у наступному:

1. Теоретичний аналіз підтвердив перевагу згорткових нейронних мереж над рекурентними у задачах детекції завдяки можливості паралельної обробки даних та стійкості до шуму, що дозволяє скоротити час навчання та підвищити точність.

2. На основі аналізу сформульовано вимоги до системи та обрано архітектуру YOLOv5s, яка забезпечує оптимальний баланс між швидкістю роботи та вимогами до обчислювальних ресурсів.

3. Реалізовано ієрархічний підхід до розпізнавання, який складається з двох етапів: детектування цілісної формули та подальшого розпізнавання окремих символів у межах знайденої області. Це дозволило спростити задачу для класифікатора символів шляхом усунення фонового шуму.

4. Створено та вручну розмічено унікальні набори даних: 721 зображення для детектора формул та 899 зображень для детектора символів, що пройшли верифікацію та балансування класів.

5. За результатами навчання моделей досягнуто високих показників точності: для детектора формул метрика mAP@0.5 склала 0,983, а для детектора символів — 0,898, що свідчить про ефективність запропонованого підходу.

6. Розроблено програмну систему з модульною архітектурою, яка включає алгоритми пост-обробки для відновлення структури виразу та інтерфейс телеграм-бота для взаємодії з користувачем.

7. Проведено експериментальну верифікацію, яка показала здатність системи коректно ідентифікувати складні математичні об'єкти та окремі символи з високим ступенем впевненості.

Розроблена система створює підґрунтя для подальшого розвитку інструментів автоматизованого введення математичних даних та може бути інтегрована в навчальні або наукові програмні комплекси.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. J. Redmon et al. "You Only Look Once: Unified, Real-Time Object Detection." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 779-788. https://doi.org/10.1109/cvpr.2016.91.
2. Y. LeCun et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86 (1998): 2278-2324. https://doi.org/10.1109/5.726791.
3. S. Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." IEEE Transactions on Pattern Analysis and Machine Intelligence, 39 (2017): 1137-1149. https://doi.org/10.1109/tpami.2016.2577031.
4. Bochkovskiy et al. "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv preprint arXiv:2004.10934 (2020). https://doi.org/10.48550/arXiv.2004.10934.
5. T. Y. Lin et al. "Focal Loss for Dense Object Detection." 2017 IEEE International Conference on Computer Vision (ICCV) (2017): 2980-2988. https://doi.org/10.1109/iccv.2017.324.
6. J. Zhang et al. "Watch, Attend and Parse: An End-to-End Neural Network Models for Handwritten Mathematical Expression Recognition." Pattern Recognition, 71 (2017): 196-206. https://doi.org/10.1016/j.patcog.2017.06.017.
7. Y. Deng et al. "Image-to-Markup Generation with Coarse-to-Fine Attention." International Conference on Machine Learning (ICML) (2017): 980-989. https://proceedings.mlr.press/v70/deng17a.html.
8. S. Elfwing et al. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning." Neural Networks, 107 (2018): 3-11. https://doi.org/10.1016/j.neunet.2017.12.012.
9. S. Hochreiter et al. "Long Short-Term Memory." Neural Computation, 9 (1997): 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

10. R. Zanibbi et al. "Recognition of Mathematical Expressions." Handbook of Character Recognition and Document Image Analysis (2014): 431-457. https://doi.org/10.1142/9789814405367_0019.

11. K. He et al. "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 770-778. https://doi.org/10.1109/cvpr.2016.90.

12. D. P. Kingma et al. "Adam: A Method for Stochastic Optimization." International Conference on Learning Representations (ICLR) (2015). https://doi.org/10.48550/arXiv.1412.6980.

13. Shorten et al. "A survey on Image Data Augmentation for Deep Learning." Journal of Big Data, 6 (2019): 1-48. https://doi.org/10.1186/s40537-019-0197-0.

14. W. Liu et al. "SSD: Single Shot MultiBox Detector." European Conference on Computer Vision (ECCV) (2016): 21-37. https://doi.org/10.1007/978-3-319-46448-0_2.

15. H. Mouchère et al. "ICFHR 2016 Competition on Handwritten Mathematical Expression Recognition (CROHME 2016)." 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR) (2016): 591-596. https://doi.org/10.1109/icfhr.2016.0113.

16. S. A. Aly et al. "Offline Handwritten Mathematical Expression Recognition: A Survey." IEEE Access, 8 (2020): 37042-37058. https://doi.org/10.1109/access.2020.2974643.

17. X. Zhang et al. "A Tree-Structured Offline Handwritten Mathematical Expression Recognition via Deep Graph Convolutional Network." IEEE Transactions on Multimedia, 24 (2022): 2153-2165. https://doi.org/10.1109/tmm.2021.3076115.

18. T. Y. Lin et al. "Microsoft COCO: Common Objects in Context." European Conference on Computer Vision (ECCV) (2014): 740-755. https://doi.org/10.1007/978-3-319-10602-1_48.

19. J. Wu et al. "Image-to-LaTeX: Learning to Convert Images to LaTeX Code." IEEE Access, 9 (2021): 80693-80703. https://doi.org/10.1109/ACCESS.2021.3084347.

20. R. Padilla et al. "A Survey on Performance Metrics for Object-Detection Algorithms." 2020 International Conference on Systems, Signals and Image Processing (IWSSIP) (2020): 237-242. https://doi.org/10.1109/iwssip48289.2020.9145130.

Додаток А

**ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ**

(Презентація)